PATENT ABSTRACTS OF JAPAN

(11)Publication number : 08-316838

(43)Date of publication of application : 29.11.1996

----

(51)Int.Cl. H03M 7/00
G06T 1/20
H04N 1/41
H04N 7/015
H04N 7/24

----

(21)Application number : 07-266750 (71)Applicant : DISCOVISION ASSOC

(22)Date of filing : 13.09.1995 (72)Inventor : WISE ADRIAN PHILIP
FINCH HELEN ROSEMARY
ROBBINS WILLIAM PHILIP
BOYD KEVIN JAMES
MARTIN WILLIAM SOTHERAN

----

(30)Priority
Priority number : 94 9405914
95 9504019
Priority date : 24.03.1994
28.02.1995
Priority country : GB
GB

----

(54) START CODE DETECTOR

(57)Abstract:
PROBLEM TO BE SOLVED: To improve extension method and device to be functioned, so as to decode/extend plural different encoded input signals.
SOLUTION: A multi-specification video decoding device has plural processing stages 36 connected by two-wire interface and arranged as pipeline processors. A control token and a data token are sent via two-wire interfaces 31, 42 to transmit both of control and data by a token format. A token decoding circuit 33 is arranged in a specific state, in order to recognize a specific token as a control token inherent in the specific stage and send an unrecognized control token along a pipeline. A reconfiguring processing circuit reconstitutes a selected stage, in response to the recognized control token arranged on the selected stage and handles only the recognized and identified data token.

Drawings are not displayable due to the volume of the data (more than 200 drawings).

CLAIMS

[Claim(s)]
[Claim 1]A machine which treats two or more bit streams which were arranged as an in-series bit stream of a digital bit characterized by comprising the following, and were coded independently, and has data carried by a pair of a start code coded independently, and said in-series bit stream.
A start code detector which has the 1st, 2nd, and 3rd stage connected in series is provided, The 1st decode means that identifies a start code relevant to said value

which each of said register memorizes a number of bits which are different from a bit stream, and said 1st register memorizes one value, and is contained in said 2nd register and said 1st register.
A circuit means which shifts said value to an end said 3rd register was beforehand decided to be.
The 2nd decode means arranged in order to receive data in parallel from said 3rd register.

[Claim 2]Two or more bit streams which were arranged as a single in-series bit stream of a digital bit characterized by comprising the following, and were coded independently are treated, A pipeline processing machine which uses two or more stages by which have corresponding data carried by a pair of a control code coded independently, and said in-series bit stream, and interconnection was carried out with 2 line interface.
A start code detector which operates according to a single in-series bit stream for generating a control token and a data token and supplying said 2 line interface.
It is a token decode circuit for having been arranged in inside of some of said stages for recognizing some of said tokens as a control token related to the stage, and passing a control token which is not recognized along with a pipeline.
Decoding which can be reconstructed and a purser processing means of operating according to a control token as which it has been recognized for reconstructing a stage specific in order to treat an identified data token.

[Claim 3]In a pipeline processing machine which has the processing stage in which interconnection was carried out by 2 line interface bus, and in which two or more reconstruction is possible, One stage in said processing stage is a space decoder, and the 2nd stage of said stages is a token generation machine which generates a control token and a data token and is passed through said 2 line interface, Some of said tokens are recognized as a control token about said space decoder, A pipeline processing machine possessing a token decode means arranged in said space decoder for constituting said space decoder which decodes said data token spatially to the 1st decoding format according to said control token.
[Claim 4]Two or more bit streams which were arranged as an in-series bit stream of a single digital bit characterized by comprising the following, and were coded independently are treated, A pipeline processing machine which uses two or more stages by which have corresponding data carried by a pair of a control code coded independently, and said in-series bit stream, and interconnection is carried out with 2 line interface.
A start code detector which operates according to a single in-series bit stream for generating a control token and a data token and supplying said 2 line interface.

A token decode means arranged in inside of some of said stages which recognize some of said tokens as a control token about the stage, and pass a control token which is not recognized through said pipeline.
It is a time decoder which operates according to a control token as which it has been recognized for processing a data token identified by reconstructing said time decoder stage and which can be reconstructed, Said time decoder which data lets said 2 line interface pass, and is moved by an 8X8-pixel data block into said time decoder.
An address means which stores and takes out said block along a block border.

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Industrial Application]This invention relates to the system of the extension method and device which function as decoding and/or elongating several different coding input signals. The example indicated below is chosen as an explanatory thing, and is related with coding in the standard of two or more numerals images. Especially this example is related with one decryption of the publicly known standard known as JPEG, MPEG, and H.261.
[0002]It is a gestalt of a control token and a data token about the interface token of an interaction (interactive) peculiar [the serial pipeline processing method of this invention], and special, It comprises one 2 line bus used in order to carry out to two or more adaptation expansion circuits which are positioned as a pipeline processor which can be reconstructed.
[0003]
[Description of the Prior Art]One of the conventional technologies has some which are written in the U.S. Pat. No. 5,216,724 specification. This device comprises two or more calculation modules, and a total of four calculation modules are connected in parallel in the desirable example. Each calculation module comprises a processing unit, dual port memory, scratch pad memory, and an Arbitration mechanism. The 1st bus connects these calculation modules and host processors. This device has the shared memory connected to the host processor and the calculation module via the 2nd bus.
[0004]The full format for transmission by which is recorded on a compact disk medium by the 4,785,349th specification of an United States patent, and a double sign is carried out to it at the conventional video frame speed is indicated. During compression, the field of a frame is separately analyzed, in order to choose the optimal fill coding method peculiar to each field. Field decryption time is presumed and a compression threshold is optimized. Grouping of the field descriptor code which

shows area size and a position is carried out by the 1st segment of a data stream. Grouping of the field philharmonic code which shows the pixel amplitude of a field is carried out by the kind of fill code, and it is placed by the segment of others of a data stream. Each data stream segment has the variable length coded according to each statistical distribution, and formatting is carried out for data frame formation. The number of bytes of a frame decreases by addition of the ancillary data determined by reverse frame sequence analysis, and the mean number chosen in order to make a halt under playback of a compact disk into the minimum is provided. This avoids the seek mode standby time characteristic [**********] of a compact disk. A decoder contains the variable-length decoder which answers the statistical information on the code stream for carrying out the variable length code of each segment of a data stream independently. Region data is obtained from field descriptive data, and is supplied with a field philharmonic code to two or more field specific decoders chosen by detecting the kind (for example, a relative value, an absolute value, the 2nd paragraph, and DPCM) of fill code. And the decoded field pixel is stored in a next display with a bit map gestalt.

[0005]The method of reducing the image data for digital television signals to the 4,922,341st specification of an United States patent using a scene model is indicated. Here, the previous frame from the scene which the picture signal supplied one by one is coded, and has already been coded in time t-1 exists in a picture storing part as a standard, and the information on a frame to a frame consists of an amplification factor, a transfer coefficient, and KUADO (4) tree block construction obtained accommodative. Initialization of a system will write similarly the picture by which half tone expression was carried out as a uniform predetermined gradation value or a regular luminance value to all the pixels (pixel) in the picture storing part of the coda by the side of a transmitter, and the picture storing part of the decoder of a receiver end. By feedback to each, both the picture storing part of coda, and the picture storing part of a decoder. The contents of the picture storing part in coda and a decoder are read with the block of variable size, and it is larger than the luminance value 1, or is amplified by the coefficient below it, and to another address of this picture storing part, operates so that writing is possible. Thereby, the block of variable size is composed according to a publicly known KUADO tree structure.

[0006]Coding/decoding device of the HDTV signal are indicated by the U.S. Pat. No. 5,122,875 specification. This device includes the compression circuit which answers the quality video source signal for providing the corresponding coding word T which specifies the kind of data which coding word CW and this coding word CW of the layered structure showing the compressed video data express. The preference circuit which answers the coding words CW and T will decompose coding word CW into a high priority and a low priority coding word string, and the priority coding word string of a high rank and lower order will correspond to a compressed video data comparatively high [importance] and low to image restoration, respectively by that cause. The transmission processor which answers a high priority and a low priority coding word string forms a high priority, the high priority of a low priority coding word, and a low priority transmission block, respectively. Each transmission block contains a header, and coding word CW and an error detection check bit. Each transmission block is supplied to the front error-checking circuit which supplies additional error-checking data. Then, this high priority and low priority data are supplied to a modem, and carry out the quadrature amplitude modulation of the corresponding subcarrier for transmission.

[0007]The video signal of odd number and an even number field is independently compressed into a U.S. Pat. No. 5,146,325 specification in order by the compressed mode in a frame, and inter-frame compressed mode, and the video decoding system which decodes the compressed image data interleaved for transmission next is indicated. Odd number and an even number field are decoded independently. Even / odd number field data is substituted for odd-/even number field data [that it cannot use] by the period without the decoded effective odd-/even number field data. It may be used effective in decoding the data of an even-/odd number field independently, and substituting for data [that the data of the opposite field cannot be used] decreasing the image display standby time under the time of a system startup, and channel change.

[0008]The video signal coding system which divides the coded video data into the transmission block for signal transmission is indicated by the U.S. Pat. No. 5,168,356 specification. When the transmitted data lost or breaks, thanks to header-data offer which enables determination of the re-penetration point to a data stream, as for the transmission block format, a receiver raises the signal recovery by a receiver end. A re-penetration point becomes the maximum by providing the secondary transmission header inserted into the coding video data within a corresponding transmission block.

[0009]In order to provide one or the function beyond it of decimation (decimation), interpolation, and the plastic surgery functions, the method of processing the image data sample for the 1 field is indicated by the 5,168,375th specification of an United States patent. This is realized by the array conversion processor etc. which are used for the JPEG compression system. The block of a data sample is changed by discrete even number cosine transformation (DECT) by both decimation and interpolation processing, and the number of frequency paragraphs is changed after that. In the case of decimation, the number of frequency paragraphs decreases and the matrix of the reduction size of the sample point in which inverse transformation is carried out and the original data block is shown is generated after this. In interpolation processing, the additional frequency component of a zero value is inserted in a frequency component array, and subsequent inverse transformation generates a series of expansion data samples without the increase in spectrum band region width. In the plastic surgery processing realized by the reefing or filtering accompanied by multiplication of conversion of the data in a frequency domain, and a filter kernel, the inverse transformation which generates the block of a series of a processed data sample is provided. By decreasing the number of ingredients to a linear phase filter, zero insertion is changed and carried out and the space sign of a kernel becomes the same as the sample number of a data block. The discrete odd number cosine transformation (DOCT) of a kernel matrix by which zero insertion was carried out is formed following this.

[0010]The system and method of transmitting a log map video image to a U.S. Pat. No. 5,175,617 specification via a telephone line band limit analog channel are indicated. The pixel configuration in a log map picture is in agreement with the geometric arrangement of the sensor human being's eye which a pixel concentrates on a center more. This transmitter divides a frequency band into a channel, and assigns one piece or two pixels to each channel. For example, a 3-kHz voice quality telephone line is divided into 768 channels which have a 3.9-Hz interval, respectively. Since each channel comprises two subcarriers of a rectangular phase, each channel can carry two pixels. Some channels are secured to the special calibration signals a receiver enables it to detect both the phase of an input signal, and a size. When a sensor and a pixel are directly linked with an oscillator county and a receiver can receive each channel continuously, this receiver does not have the necessity of taking a transmitter and a synchronization. A receiver takes the 1st frame and a synchronization and an FFT algorithm performs high-speed discrete approximation to the case of the continuous action which obtains a following frame for every rear frame period of the. Since a frame period is comparatively low compared with a sampling period, once the 1st frame is detected, a receiver will not lose a frame synchronization. In the experimental video telephone, four frames were transmitted to 1 second, quadrature encoding was carried out to 1440 pixel log map pictures, and the effective data-transfer rate of 40,000 bits/s or more was obtained.

[0011]The video compression system which has a video signal of the odd number and even number field where the video signal of odd number and an even number field was independently compressed into the U.S. Pat. No. 5,185,819 specification in order by the compressed mode in a frame and inter-frame compressed mode is indicated. The data in which the even number field in a frame was compressed in the odd number and even number field of data which were compressed independently is interleaved for transmission so that it may generate while being between the continuous fields where the odd number field in a frame was compressed. For a receiver, the penetration point size to the signal for coding doubles by a series of interleaved fields, without increasing a transmitted data amount.

[0012]The device and method of processing a video data in real time for compression/decoding are indicated by the U.S. Pat. No. 5,212,742 specification. This

device comprises two or more calculation modules, and a total of four calculation modules are connected in parallel in the desirable example. Each calculation module comprises a processing unit, dual port memory, scratch pad memory, and an Arbitration mechanism. The 1st bus connects these calculation modules and host processors. At the last, this device has the shared memory connected to the host processor and the calculation module via the 2nd bus. This method performs partial cleavage reliance of a picture, in order that each processing unit may process.
[0013]suitable for use of the ISO/IECMPEG standard proposed by the U.S. Pat. No. 5,231,484 specification -- it is indicated. Three components or subsystem which cooperates is contained in this. In order to provide the optimal vision quality in consideration of the number of bits assigned to the picture, This three components or subsystem pretreats the digital animation sequence inputted on a variable adaptation target, assigns a bit to a picture in order, and quantizes a conversion factor accommodative in order of video to the field to which pictures differ.
[0014]The method of removing the frame relative redundancy of the animation sequence in a computer system is indicated by the U.S. Pat. No. 5,267,334 specification. This method comprises the process of detecting the 1st scene change of an animation sequence, and the process of generating the 1st key-frame including scene information perfect for the 1st picture. In a desirable example, this 1st key-frame is known as a "positive" key-frame or intra-frame, and usually exists in a CCITT conformity compressed video data. This processing possesses the process of generating at least one middle compression frames, and one middle compression frames include difference information with the 1st picture for this few **** to at least one picture which follows the 1st picture in time within said animation sequence. This at least one picture is known as inter-frame. There is a process of generating the 2nd key-frame including scene information perfect for the picture displayed on the last as the process of detecting the 2nd scene change in an animation sequence just before the 2nd scene change. This 2nd key-frame is known as a "backward" key-frame. Said 1st key-frame and at least one middle compression frames are connected for forward direction reproduction, and said 2nd key-frame and said middle compression frames are connected for reverse direction reproduction. When forward direction reproduction of the picture is carried out, said intra-frame one may be used for generating perfect scene information. When reverse reproduction of this scene is carried out, a back key-frame is used for reproducing perfect scene information.
[0015]The 1st circuit apparatus that changes from publicly known picture and pyramid stage of a predetermined number to a U.S. Pat. No. 5,276,513 specification is indicated with the 2nd circuit apparatus that comprises an animation and a vector stage with new said predetermined number and same number. These devices are real time and conduct high hierarchical animation analysis (HMA) of a cost effect by the minimum hardware structure, using the minimum system processing delay and/or the minimum

system processing delay. A high-resolution image data is comparatively answered from the continuous input sequence of the continuous predetermined picture-element-density picture data frame especially generated in a comparatively high frame speed (for example, speed per second 30 frames), Said 1st and 2nd circuit apparatus acquire the continuous output sequence of the continuous predetermined picture-element-density vector data generated in the same frame speed after a certain processing-system delay. Each vector data frame shows the motion of the picture which happens by the consecutive image inter-frame of each set.
[0016]On U.S. Pat. No. 5,283,646 specifications. The method and device which update the quantization step size used for quantizing the coefficient showing the picture which makes send out the bit of a desired number precisely for every frame, for example, is transmitted via a communication channel are indicated coding a picture only once to a real-time video coding system. This data is divided into the sector which has two or more blocks. It is coded, for example using DCT encoding, and this block generates a coefficient sequence for every block. This coefficient is quantized, and if the number of bits taken to describe data depends on a quantization step, it is changed considerably. At the time of the end of transmission of the data of each sector, the used actual accumulation number of bits is compared with the accumulation number of bits of the request used to a selected number relevant to this particular group's data of sectors. This system readjusts quantization step size, in order to obtain the number of data bits of a final request to two or more sectors with which a picture is expressed, for example. Various methods are indicated, in order to carry out re-new [of the quantization step size] and to determine desired bit allocation.
[0017]Yong M. of waste-cloth contest technical paper (Wescon Technical Papers) No. October, 1984 [2 or]/, and the November item -- jon (Chong, Yong M) -- the paper of work. The signal processing system in the real time designed by image processing is indicated by especially "the data flow architecture of digital image processing" (A Data-Flow Architecture for Digital Image Processing). The data flow architecture which specifically used as the base the token which is fixed 1 word length who has a fixed-length address field is indicated. The token comprises a data field, a control field, and a tag. The tag field of a token is further divided into a processor address field and an identifier field. Since processor address feeding sends out a token to a data processor surely, it is used, and since an identifier field is told what kind of processing should be performed to a data processor, it is used for the purpose of attaching a label to data. Thus, an identifier field acts as directions to a data processor. A system sends each token to a specific data flow processor using a module number (MN). When this MN is in agreement with MN of a special stage, a suitable operation is performed to data. When not in agreement, a token is sent to an output data bus.
[0018]Vo1.23 of an IEEE J solid state circuit (Solid-State Circuits), and the No.

February, 1988 [1 or] item, In the "elastic pipeline mechanism by a self-adapting circuit" (An Elastic Pipeline Mechanism by Self-Timed Circuits) of work besides KIMORI (Kimori), the elastic pipeline who has a self-adapting circuit is indicated. An asynchronous pipeline possesses two or more pipeline stages. Each pipeline stage is provided following the combination logic circuit which performs a logical operation peculiar to a pipeline stage. It is constituted by the input day TARRACHI group. The trigger signal generated from the data transfer control circuit accompanying this pipeline stage is simultaneously supplied to a data latch. A data transfer control circuit is connected mutually, forms a chain, and is controlling the data transfer between latter pipelines by the transmission signal line and the ACK signal line in handshake mode via this. In order to choose the operation performed to the operand of the present pipeline stage, the decoder is usually provided in each stage. In order to carry out code processing of the complicated decoding processing beforehand, and in order to reduce the problem of serious I course for a logic circuit, a decoder can also be arranged in the preceding paragraph. Since it opts for the interaction of a sub inter module based on the determination localized thoroughly and each submodule can perform data transfer for a data buffer ring and self-adapting data transfer simultaneously uniquely, any concentrated control is removed by a pipeline's pliability. Finally, since a pipeline's pliability is increased, the empty pipeline stage was provided between the pipeline stages occupied, and data transfer with the reliability between each stage has been secured.
[0019]It is selectable in the number of the coefficients contained in each subblock, and, as for a U.S. Pat. No. 5,278,646 specification, the code which shows the number of the coefficients in each layer is indicating the improved decoding art which each coding video sequence begins and by which bit stream insertion is carried out. According to this art, the original run of the zero coefficient in a most high-ranking resolution (resolution) layer can be made eternal by forming the subblock about each scale along with a continuation scan from a selected number of coefficients. These subblocks are decrypted by canonical form voice by applying a reverse discrete cosine transform to the subblock of the square obtained suitable zero embedding (padding) or by discarding an excessive coefficient from each scale. This art gives a still clearer end-of-block signal to a separating block, and, in most cases, improves decoding efficiency by making it unnecessary to decode a clear end-of-block signal.
[0020]A U.S. Pat. No. 4,903,018 specification indicates the method of compressing / elongating the structurally related multiplex data sequence, and a data processing system. In order to show the characteristic common to the data element in which the predetermined number in a data sequence continues, analysis is peculiar to the data set currently made about structure. Instead of a data element, the code decrypted again is used between extension. The common characteristic is called for by analyzing the data element which has the same degree in the number of data sequences. the

data element obtained by decrypting a code during expansive is based on the degree of these data series based on the degree of these data elements -- the inside of data series -- grade attachment ****. The data processing system which performs this method possesses the index storing 28 which has a line address of the storing matrix 26 in the line sequence relevant to the storing matrix 26.
[0021]A U.S. Pat. No. 4,334,246 specification indicates the circuit which resembles the compression which it had the front for transmission and storing, then compresses video, and a method. Here, it operates by the one-shot prediction device of two lines, and is coded by 4, 8, or the 12-bit code word using run length coding, and the original video generated by the raster input scanner is packed to 1 bit data word. And an expandor combines 16 bit data word and unpacks data, Separate into each code word and all change a code word into the nibble of a large number which are 4 bits of 0, The termination of 1 which constitutes decoding data, or the nibble which has two or more bits 1 is carried out, In order to generate a DEPUREDIKUTA bit, the actual video of the line under present scan and the video bit before the present line are investigated, and in order to generate final actual video, decoding data and a DEPUREDIKUTA bit are measured.
[0022]A U.S. Pat. No. 5,060,242 specification carries out the DPCM coding of the signal, in order to generate the variable-length code word [be / in order not to find any losses of data and to raise transmission efficiency / no crevice] packed firmly, and it indicates Huffman and the image signal processor which carries out run length coding for a signal. The device which processes this word packed firmly has the barrel shifter (barrel shifter), and that shift modulus is controlled by accumulator receiving code word length information. An OR gate is connected to a shifter and a register is connected to a gate. The device which processes this digital signal that packed firmly and abolished correlation possesses coda by the barrel shifter for unpacking, an accumulator, the Huffman decoder and a run length decoder, and reverse DCPM.
[0023]In order to provide one of the decimation attained using an array conversion processor which is adopted by the JPEG compression system, interpolation, and the sharpening, or two or more functions, a U.S. Pat. No. 5,168,375 specification, The method of processing the field of an image data sample is indicated. The block of a data sample is changed by discrete even number cosine transformation (DECT) in both decimation and interpolation processing. Then, the number of frequency paragraphs is changed. The number of frequency paragraphs decreases, and in order to generate the sample point showing an original data block of the reduced size after that, inverse transformation of the case of decimation is carried out. In interpolation, the additional frequency component which has a zero value is inserted in the array of a frequency component. Subsequent inverse transformation generates the data sampling set expanded without increasing a spectrum zone. In the case of the convolution containing the multiplication of conversion of data and the filter kernel in

a frequency domain, or sharpening realized by filtering operation, the inverse transformation processing attributed to the block set of a processed data sample is included. It is overlapped with a block, the specified sample is saved, and an excessive sample is thrown away from areas of overlap. Space expression of the kernel about a linear phase filter is changed by decreasing the number of ingredients. Zero are embedded in order to make the sample number of a data block equal. Then, discrete odd number cosine transformation of the embedded kernel matrix is performed.

[0024]A U.S. Pat. No. 5,231,486 specification indicates the high resolution video system which processes the bit stream containing the priority variable-length-coding data word of a high level and a low. Coded data is divided into a high ply cage TIDETA pack and a low priority data pack by each data packing unit. Coded data is continued and supplied to both packing units. The high priority length data and low priority length data showing the high priority ingredient length of coded data and low priority ingredient length are supplied to a high ply cage TIDETA packer and a low priority data packer, respectively. Since high ply cage TIDETA is outputted via the 1st output path, when you pack, let low priority data be zero. Since low priority data is outputted via the 2nd output path, when you pack, let high ply cage TIDETA be zero.

[0025]A U.S. Pat. No. 5,287,178 specification indicates a video signal coding system including the signal processing part which divides a coding video data into the transport blocks which have a header unit and a data pack part. In order to enabling-ize signal processing which begins the described sequence, in the described un-simultaneous phase sequence, the reset control circuit which cancels reset of a system component after global reset also possesses this system. That the data line is transmitted to valid data will start a phase reset release sequence, if detected.

[0026]A U.S. Pat. No. 5,124,790 (Nakayama) specification indicates the inverse quantization device used with image memory. An inverse quantization device is used depending on how to use usual, in order to decode difference prediction-coding (DPCM) code data.

[0027]A U.S. Pat. No. 5,136,371 (Savatier et al.) specification is determined by the full state of a buffer, and is related with the inverse quantization device which has an adjustable quantization level which is variable. The feature of this conventional technology is the ability to attain the greatest adjustable data rate. Overflow of a buffer or underflow is avoidable by adapting the quantization step size of the quantizer 152 and the inverse quantization device 156 using the quantization level re-calculated after each block is coded. A quantization level is calculated as a function of the quantity of the data already coded about the frame to total buffer size. Thus, a quantization level is preferably re-calculated by the decoder and is not transmitted.

[0028]A U.S. Pat. No. 5,142,380 (Sakagami et al.) specification indicates the image compression device about a still picture which was photoed by the electronic "still" camera which used the solid state image pickup device. Here, the quantizer used is

connected to ROM15 which stores the threshold of the memory measure which stores the threshold of the quantizing matrix about the luminance signal Y, and the quantizing matrix about chrominance signal I and Q.
[0029]A U.S. Pat. No. 5,193,002 (Guichard et al.) specification indicates the device which codes / decrypts a picture signal in real time in relation to CCITT standard H.261. A digital signal processor performs direct quantization and inverse quantization.
[0030]A U.S. Pat. No. 5,241,383 (Chen et al.) specification indicates the false constant bit rate video coding device realized by adjusting a quantization parameter. The quantization parameter used with the quantizer 32 is periodically adjusted, in order to increase or decrease the quantity of the code bit generated by the coding circuit. The variation of the quantization parameter for coding the next picture group is decided with the deviation measured between the actual numbers of code bits generated by the coding circuit to the picture group of before in a former picture group's number of presumed code bits. The number of code bits generated by the coding circuit is controlled by controlling quantization step size. Generally, small quantization step size brings about many numbers of code bits, and big quantization step size brings about the small number of code bits.
[0031]A U.S. Pat. No. 5,113,255 specification (Nagata et al.), A U.S. Pat. No. 5,126,842 specification (Andrews et al.), A U.S. Pat. No. 5,253,058 specification (Gharavi), a U.S. Pat. No. 5,260,782 specification (Hui), and a U.S. Pat. No. 5,212,742 specification (Normile et al.) are quoted for explanation of a background art and general art.
[0032]
[Problem(s) to be Solved by the Invention]In the pipe-line system which has two or more processing stages which this invention has between an input, an output, and said input and an output, Interconnection of two or more processing stages is carried out by the interface of the 2-wire system which transmits a token along with a pipeline, A control token and a data token interface with all the processing stages in a pipeline, It is a gestalt of the universal adaptation unit which interacts with the selected processing stage in a pipeline, and a pipeline's processing stage is related with the improved pipe-line system, wherein the pliability which improved in composition and processing is given. According to this invention, a processing stage is constituted so that recognition of at least one token may be answered. One processing stage is a start code detector which generates / changes a token in response to an input.
[0033]The picture start code token, as for the token of this invention, the start of a picture instructs back Lycium chinense to be to the following data token, The picture which directs the end of each picture and a token, the flash plate token which clears a buffer and resets a system, In order to process according to one as which it was chosen of two or more picture compression / extension standards, it is a coding standard token which adjusts a system. This invention The Huffman decoder, and an index / data (ITOD) stage, An arithmetic logic unit (ALU) and the data buffer means

which continues immediately after a system are provided, and it is related with the pipe-line system for the decryption of a video data with a controllable time interval for the video picture from which data size changes. According to this invention, a processing stage receives an input data stream, a means to recognize a specific bit stream pattern is provided, and a processing stage makes random access and error recovery easy. This invention attains a clear end to picture data decoding, points to the end of a picture, and possesses a means to perform picture stop after operation which clears a pipeline. The improved pipe-line system possesses a means to pad a buffer in order to pass arbitrary numbers of bits through the buffer of fixed size and fixed width, and a buffer. In order that this invention may have further a data stream containing a run length code and may extend a run-levels code to one run of the zero data which follows with one level, The reverse modeler means operation-ized by the data stream from a token is provided, and each token is expressed by a specified number of values. This invention is arranged between a reverse modeler stage, a reverse discrete cosine transform stage, and a reverse modeler stage and a reverse discrete cosine transform stage, and possesses the processing stage which answers the token table for data processing.
[0034]Provide this invention and the Huffman decode means which decodes the data word coded based on H.261, JPEG, or the Huffman code-ized conditions of one of MPEG standards in the improved pipe-line system data word, A means to receive Huffman code-ized data word further including the identifier which identifies the Huffman code standard by which data word was coded based on it, A means to read an identifier in order to determine which standard managed Huffman code-ization of the received data word, Reading of the identifier which identifies Huffman code-ized data word as it is H.261 or MPEG Huffman code-ized settled is answered if needed, A means to be connected to a Huffman code-ized data word reception means so that an operation is possible, and to receive an index from an index creating means, and to generate the index relevant to each JPEG Huffman code-ized data word, A means to output the data word which operated the table for reference including a Huffman code table with the format used based on the JPEG standard in order to transmit JPEG Huffman expression news, and was decoded corresponding to the received index is provided.
[0035]
[Means for Solving the Problem]In a pipe-line system which, simply and generally, has two or more processing stages which this invention has between an input, an output, this input, and an output, In order to transmit a token along with a pipeline, interconnection of the processing stage is carried out by 2 line interface, A control token and a data token interface with all the processing stages in a pipeline, it is a gestalt of a universal adaptation unit which interacts with a selected processing stage in a pipeline, and a processing stage is given pliability which improved in composition

and processing.
[0036]Each processing stage in a pipeline may have both first and the second memory measure. A processing stage is reconstructed by selected token according to recognition. A pipeline's token is located regardless of a processing stage which is dynamically accommodative, and is located according to a processing stage which performs a function, or performs a function.
[0037]In a pipeline of this invention, a token may be changed by interfacing with a stage. A token may interact with all the stages of a pipeline, or may interact only with specific stages fewer than all. A token in a pipeline may interact with a stage which may interact with an adjoining stage or does not adjoin. A token reconstructs a processing stage. Such a token may be subordinate to a certain function, may be located, and may be located regardless of other functions.
[0038]A token may show further that an additional word exists in this token, and may also contain an extension bit which identifies the last word in this token. Variable length may be sufficient as an address field of a token, and the Hoffmann coding is carried out typically.
[0039]A token is generated by processing stage. A pipeline token may contain data transmitted to a processing stage, and the token may not have data. A certain token is identified as a data token, and gives data to a pipeline's processing stage. Other tokens are identified as a control token and adjust a processing stage. This adjustment includes reconstruction of a processing stage. Other tokens carry out both supply of data, and adjustment to a processing stage. Either of the tokens identifies a coding standard to a pipeline's processing stage. Between processing stages, other tokens are not dependent on any coding standards, and operate. A continuous change by a processing stage is also possible for a token.
[0040]The pliability of an interaction of a token which cooperates with a processing stage makes easy big functional distribution of a processing stage to an internal structure, system extensiveness is carried out and the pliability of a token makes a change easy. About this point, the token can also make easy two or more functions in a processing stage. A token may be based on hardware or may be based on software. A token carries out control to offer simultaneously with data in a processing stage.
[0041]According to this invention, this invention is transmitted into a single bit stream, In order that a control code and corresponding data may have the pair coded, respectively and may treat two or more encoded bit streams as a serial bit stream of a digital bit, A pipe-line system to which interconnection of two or more processing stages was carried out by 2 line interface is characterized by comprising:
A start code detection means which answers a single serial bit stream, generates a control token and a data token, and is given to 2 line interface.
A token decode means which is provided in a predetermined processing stage, recognizes a token as a control token peculiar to this stage, and passes an

unrecognized control token along with a pipeline.
A reconstruction decoding parser processing means to answer a recognized control token and to reconstruct a specific stage in order to treat an identified data token.

[0042]According to this invention, one stage is the start code detector constituted so that an input might be received and a token might be generated / changed, A start code detector answers data, generate a token, and a start code is searched, It detects, and the result is answered, a token is generated, a duplication start token can be detected, the 1st start code is disregarded, and it is used for generation of the 2nd start code of a start code token.
[0043]A start code detector stage searches input data with search mode, and finds a selected start code. A detector searches discontinuation in an input data stream, and search is performed to data from an external data source. A start code detector stage generates a start code token, a picture start token, a slice start token, a picture and a token, a sequence start token, a sequence and a token, and/or a group start token. A start code detector stage performs a padding function which embeds a bit at a word of the last of a token.
[0044]A start code detector is provided with the following.
Two or more bit streams which were arranged as an in-series bit stream of a digital bit, and were coded independently are treated, And in a machine which has data carried by a pair of a start code coded independently, and said in-series bit stream, A start code detector subsystem which has the 1st, 2nd, and 3rd stage by which the series connection was carried out is provided, The 1st decode means that identifies a start code relevant to a value which each of a register memorizes a number of bits which are different from a bit stream, and the 1st register memorizes one value, and is contained in the 2nd register and the 1st register.
A circuit means which shifts this value to an end the 3rd register was beforehand decided to be.
The 2nd decode means arranged in order to receive data in parallel from the 3rd register.

[0045]The 2nd decode means is answered and 1 memorized as a result of decoding of a value relevant to a start code or a memory which outputs two or more control tokens is provided. Two or more tag shift registers treating a tag in which the validity of data of register-izing is shown are provided. A system possesses further a means to access an input data stream of formation of a microprocessor interface, and a means which formats / systematizes a data stream.
[0046]According to this invention, a start code detector identifies a start code of variable width relevant to a bit stream decrypted by differing. A detector generates two or more data tokens from an input data stream. A system is a pipe-line system

and a start code detector is arranged as the 1st stage of a pipe-line system.
[0047]This invention possesses an omnipotent adaptation unit of a gestalt of an interaction interface token for a control facility between two or more processing stages, and/or a data function, A token provides a system which is a picture start code token which directs that a start of a picture follows in the following data token.
[0048]A token is a picture and a token which direct an end of each picture.
[0049]A token is a flash plate token which clears a buffer and advances a system downstream from an input to an output and which takes and resets a system. A flash plate token resets a stage variably as it advances a pipeline to the downstream. In order that a token may process according to one as which it was chosen of the picture compression / extension standards of the number of **, it is a coding standard token which adjusts a system.
[0050]A coding standard token directs JPEG and/or other suitable standards as a picture standard. Some processing stages answer a coding standard token, and are reconstructed. [at least]
[0051]One of the processing stages of a system is the Huffman decoder parser, a coding standard token is answered and a parser is reset by the address position corresponding to a position of a program treating a picture standard specified by coding standard token. It is selectable by a coding standard control token corresponding to a memory location used in order that a reset address may test the Huffman decoder parser.
[0052]It is shown whether the Huffman decoder possessed a decoding stage, and an index / data stage, in order that a parser stage might choose a table required for a coding standard identified specially, directions were sent to the index / data stage, and data which acted as Irikita was reversed.
[0053]A token mentioned above may be an interaction meta-MORUFIKU interface token.
[0054]This invention possesses the Huffman decoder, an index/data unit, an arithmetic logic unit, and a data buffer means that continues immediately after a system, and provides a video-data decoding system with a controllable time interval for a video picture from which data size changes.
[0055]This invention possesses 2 line interface which carries out interconnection of the processing stage, and an interface provides a space decoder which makes possible parallel processing about serial processing about data, and control.
[0056]As mentioned above, a system of this invention possesses ROM with an individual program memorized to each of two or more picture standards, by a token, it is selectable and a program makes easy implementation of processing to two or more picture standards.
[0057]A space decoder system of this invention possesses a token formatter which formats a token, and, as a result, a data token is created.

[0058]A system of this invention possesses a decoding stage and a parser stage which sends an instruction for choosing a required table to an identified specific coding standard to an index/data unit, and a parser stage directs whether data which carries out ingress is reversed. A table is arranged in a suitable table in a memory which makes multiple use possible.
[0059]This invention has an input data stream, a processing stage which receives an input data stream is provided, and a processing stage possesses a means to recognize a specified bit stream pattern, and provides a pipe-line system with easy random access and error recovery. According to this invention, a processing stage is a start code detector and a bit stream pattern contains a start code. Therefore, this invention possesses a search mode means to search for a different encoding data stream arranged as a single in-series data stream, in order to perform random access and strengthened error recovery.
[0060]This invention performs picture after stop operation, in order to realize a clear end to picture data decoding, It points to a picture end, a means to clear a pipeline is provided, and this means provides a pipeline machine which generates combination of a picture and a token, and a flash plate token.
[0061]Providing a means to pad a buffer, in order for a pipeline machine of this invention to pass arbitrary numbers of bits through a buffer of fixed size and fixed width, and a buffer, a padding means is a start code detector.
[0062]Padding is performed to a word of the last of a token and padding ensures homogeneity of wordsize. According to this invention, a reconstruction stage is provided as a space decoder and a padding means adds sufficient overhead bit which has the bit length with each same extension picture in an output of a space decoder to picture data currently processed by space decoder.
[0063]In a system which has a data stream in which this invention contains a run length code, It is operation-ized by data stream from a token, and a system which possesses a reverse modeler means to extend a run-levels code to one run of zero data which follows with one level and by which each token is expressed by a specified number of values is provided.
[0064]A reverse modeler means intercepts a token which lacks a value of a predetermined number, and predetermined numbers are 64 coefficients in the desirable example.
[0065]A device which carries out this invention accepts a data token which has a run length code, and possesses an expansion circuit which decodes a run length code. It judges whether a padder circuit has the length predetermined in a data token by communicating with an expansion circuit, and a data unit is added to a data token until it reaches predetermined length, when a data token is shorter than predetermined length. A bypass circuit makes an expansion circuit and a padder circuit avoid tokens other than a data token.

[0066]In a method of according to this invention processing data in order to fill a buffer efficiently, It has at least one of the following formats, : which provides a token of the 1st type with the 1st prescribed width -- format A - ExxxxxxLLLLLLLLLLL format B - ERRRRRRLLLLLLLLLLL format C - E -- 000000 LLLLLLLLLLL here, They are E= extension bit, F= specific format, R= run bit, L= length bit or a non-data token, and x= "boss TOKEA" bit.
[0067]A format A token is divided into a format 0a token of a form of ELLLLLLLLLLL, A format B token is divided into format 1 token of a form of FRRRRRR00000, and a format 0a data token, A format C token is divided into format 0 token of a form of FLLLLLLLLLLL, and format 0, format 0a, and format 1 token is packed to a buffer which has the 2nd prescribed width.
[0068]In a device for this invention to supply a time delay to a group of a compressed picture corresponding to video compression / extension standard, Data word containing a compressed picture counts by a counting circuit, it communicates with a counting circuit, start information is received according to a standard about video extension, and a microprocessor which transmits start information to a counting circuit is provided.
[0069]Data word is accepted, a reverse modeler circuit delayed in data word communicates with a counting circuit and a control circuit which exists in the middle of a reverse modeler circuit, and a control circuit compares start information with a counted word of data, and signs a control circuit. A control circuit puts a signal corresponding to data word which suited a start-up standard into queuing, and controls a reverse modeler delay function.
[0070]In a pipe-line system which has a reverse modeler stage and a reverse discrete cosine transform stage, this invention is arranged between a reverse modeler stage and a reverse discrete cosine transform stage, and is characterized by a processing stage which answers a token table for data processing.
[0071]A token of this invention is a quantization table token which generates a quantization table from a processing stage.
[0072]Provide this invention and a means to decode data word coded based on H.261, JPEG, or Huffman code-ized conditions of one of MPEG standards data word, Provide a means to receive Huffman code-ized data word, including an identifier which identifies a Huffman code standard by which data word was coded based on it, and a receipt means, A means to read an identifier which determines which standard managed Huffman code-ization of received data word, Reading of an identifier which identifies Huffman code-ized data word as it is H.261 or MPEG Huffman code-ized settled is answered if needed, A means to change data word into data word formed into the JPEG Huffman code is provided, A means to generate an index relevant to each JPEG Huffman code-ized data word which is connected to a Huffman code-ized data word reception means so that an operation is possible, and receives an index

from an index creating means, The Huffman decoder possessing a means to output data word which operated a table for reference including a Huffman code table with a format used based on a JPEG standard in order to transmit JPEG Huffman expression news, and was decoded corresponding to a received index is also provided.
[0073]This invention H.261, JPEG, Or in a method for decoding data word coded based on Huffman code-ized conditions of one of MPEG standards, Data word possesses a process in which Huffman code-ized data word is received including an identifier which identifies a Huffman code standard coded based on it, and a process to receive, An identifier which determines which standard managed Huffman code-ization of received data word [a process and if needed] to read H.261, Or a process in which answer reading of an identifier which identifies Huffman code-ized data word as finishing [formation of an MPEG Huffman code], and data word is changed into JPEG Huffman code-ized data word is included, A process in which an index relevant to each received JPEG Huffman code-ized data word is generated, It has a process in which an index is received, and a process in which data word decoded corresponding to a received index is generated, In order to transmit JPEG Huffman expression news, a process in which a table for reference including a Huffman code table with a format used based on a JPEG standard is operated is provided.
[0074]During a statement of an example of the following this inventions, since the following term is used frequently, general explanation of the term is given to below.
[0075]Term explanation block: A matrix or 64 DCT coefficients (sauce, quantization, or inverse quantization) of eight-line the pixel of eight rows.
[0076]Chrominance (ingredient): One matrix and a block showing one of two color-difference signals concerning the three primary colors which are specified by a bit stream, or one pixel. Numerals used for a color-difference signal are Cr and Cb.
[0077]Coding notation: A data element expressed with a coding gestalt.
[0078]Coding video bit stream: Coding notation of a series of one or more pictures specified on these specifications.
[0079]Encoding-order watch: Turn which a picture is transmitted and is coded. The same necessity as display order watch does not necessarily have this turn.
[0080]Ingredient: One pixel from one of three matrices (luminosity and two chrominance) which constitute one matrix, a block, or a picture.
[0081]Compression: Decrease the number of bits used for expressing one item of data.
[0082]Decoder: An example of decoding processing.
[0083]Decryption (processing): Processing which generates a picture or a sound sample which read inputted encoded bit streams in a definition on these specifications, and was decrypted.
[0084]Display-order watch: Turn that a decryption picture is displayed. Generally, a picture of this is the same as turn of having been inputted into an encoder.
[0085]Coding (processing): Processing which generates effective encoded bit streams

which read an inputted image sequence or a sound sample, and are defined by this specification although not specified on these specifications.
[0086]Intra (intra) coding: Coding of a macro block or a picture which uses only a macro block or information from a picture.
[0087]Luminosity (ingredient): A thing concerning the three primary colors which are specified by a bit stream by one matrix and a block showing black and white of a signal, or one pixel. Numerals used for a luminance signal are Y.
[0088]Macro block : Two (for a 4:2:0 chroma format) from 16x16 portions of four 8x8-block luminance data and a brightness component of a picture. Four (for a 4:2:2 chroma format), or eight 8x8-block corresponding chrominance data (for a 4:4:4 chroma format). A macro block may say picture element data and may say that by which a data element of others which are specified to a macro block header of syntax defined as this portion of this specification as a pixel value was coded. For a person skilled in the art who has the usual knowledge, the use is clear also from the context.
[0089]Animation compression: Raise predictive efficiency of a pixel value using an animation vector. This prediction provides with offset a reference image of the past and/or the future including a pixel value decrypted before using an animation vector and using it for creation of a prediction error signal.
Animation vector: A two-dimensional vector used for animation compensation which provides offset to coordinates of a reference image from a coordinates position of the present picture.
[0090]Intra [non-] (non-intra) coding: Coding of a macro block generated at time different from a macro block or information from picture itself, a macro block which also uses information from a picture, or a picture.
[0091]pixel: -- picture unit picture: -- sauce and image data coded or reconstructed. Sauce or a reconstructed picture comprises three square matrices which have a numerical value showing luminosity and two chrominance signals of 8 bits, respectively. In video which that of a picture is the same as that of a frame, and was interlaced in advanced video, a picture points out the field of a frame or a head of the frame, or the last field by the context.
[0092]Prediction: Predict a pixel value or a data element decrypted now using a prediction device.
[0093]A processing stage (RPS) which can be reconstructed: A stage which answers a recognized token, reconstructs self and carries out various work.
[0094]Slice: A series of macro blocks.
[0095]Token: A universal adaptation unit of a gestalt of a package of an interaction interface messenger cable for control and/or a data function.
[0096]Start code (a system and video): 32-bit numerals inserted in unique encoded bit streams. These numerals are used for some purposes of having included identifying some of structures in coding syntax.

[0097]Inverse processing for coding which assigns a code word shorter than variable-length-coding (VLC): to a frequent event, or assigns a longer code word to an event which is not frequent.
[0098]Video sequence: A series of one or more pictures.
[0099]
[Function]According to this invention, it controls in two or more processing stages and a processing stage, The universal adaptation unit of the gestalt of the interaction interface control token which performs a data function is provided, and the pipe-line system given the pliability whose processing stage improved in composition and processing is provided.
[0100]
[Example]Hereafter, with reference to drawings, the example of the pipe-line system by this invention is described.
[0101]As an outline about the most general function used for the pipe-line system used for the desirable example of this invention, drawing 1 is the explanatory view simplified substantially [of six cycles of 6 stage pipeline]. (Some advantageous functions which are not shown in drawing 1 at a pipeline's desirable example are ******s so that it may explain still in detail below.)
It supposes that drawings are referred to, and whether the same reference number's being the same and a corresponding element are shown through various figures contained in a drawing, and, more particularly, drawing 1 is a block diagram of six cycles at the time of carrying out this invention. Each sequence of a box shows one cycle and each of a different stage is classified from A by F, respectively. Each box which drew the slash shows that data with an effective corresponding stage, i.e., the data which should be processed in one of the pipeline stages, is held. After processing (contained other than [no] the simple transmission without operation of data), effective data is transmitted by the pipeline as effective output data.
[0102]Note that it is more than six to actual pipeline application, or few pipeline stages may be included in it. In the case of the pipeline stage of the arbitrary number, it is usable in this invention so that he may naturally notice. If it does not interfere even if it processes one data in many stages, and stages differ, processing time may also differ.
[0103]It is adding to the clock and the data signal (it explains to a less or equal), and, as for a pipeline's, two transfer control signals, i.e., the ″VALID″ (effective) signal, and the ″ACCEPT″ (permission) signal are included. These signals are used in order to control the data transfer in a pipeline. The VALID signal illustrated as a line of the upper part of the two lines which connect the adjoining stage is supplied from each pipeline stage to a forward direction, i.e., a downstream direction, even to a nearby contiguity device. Even if this device is other pipeline stages or a certain another system, it does not interfere. For example, even if the last pipeline stage supplies the

data to the next processing circuit composition, it does not interfere. Even a front device supplies the ACCEPT signal illustrated as a line of the bottom of the two lines which connect the adjoining stage toward another upstream direction.

[0104]The data pipe-line system of the type used for realization of this invention has the characteristic beyond one or it of the characteristic shown below, as shown in a desirable example.

[0105]1. The disturbance which a pipeline originates in that is, "it being elastic" and the delay which took place in the specific pipeline stage, and is caused in other pipeline stages is the smallest possible disturbance. The continuous pipeline stage can continue processing, therefore it means producing a gap in the data flow which follows the delayed stage. Similarly, the pipeline stage to precede is also continuable as much as possible. In this case, the gap in a data stream is removed as much as possible from data flow.

[0106]2. The control signal which arbitrates a pipeline is organized so that these signals may be restricted to a nearby adjoining pipeline stage and it may spread. The stage of the above [case / of the signal which flows in the same direction as data flow] is a stage following just the back. The stage of the above [case / of the signal which flows into data flow and an opposite direction] is a stage just front.

[0107]3. The data in a pipeline is coded so that the data of the type with which a large number differ may be processed within the pipeline concerned. This coding needs to accommodate the data packet of variable size, and the size of a packet does not need to be known beforehand.

[0108]4. The overhead relevant to type description of data is small as much as possible.

[0109]5. It is possible for each pipeline stage to recognize only the data type of the minimum number required for the function required of a pipeline stage. However, even if a pipeline stage is a case where the aforementioned data type is not recognized, it can supply all these data types to the continuous stage. If this is possible, communication between the pipeline stages not adjoining will be enabled.

[0110]Although not illustrated by drawing 1, there is a data line, and even if it is any of two or more single tracks or several parallel lines, the data bus introduced or derived to each pipeline stage is formed. Data is transmitted between transference, transfer, and a stage via a data line to a pipeline's stage so that it may explain and illustrate very in detail below.

[0111]The 1st pipeline stage should be careful of being ability ready for receiving about the data and the control signal from a precedence device of arbitrary forms. The precedence device in this case means the receiving circuit of a digital image transmission system, other pipelines, etc., for example. In another side, the 1st pipeline stage can generate all or some of data processed in the pipeline concerned. So that it may explain in detail below as a practical question a "stage", although the

arbitrary processing circuits containing the system (only supply data) which nothing carries out, or all the systems (for example, other pipelines, multiplex systems, or multiplex pipelines) are included, [interfere and] Generation, change, and deletion of data are possible for a "stage" if needed. When a pipeline stage contains the effective data caudad transmitted through the pipeline concerned, the VALID signal which shows data validity does not have an important point transmitted further in the distance from the next pipeline stage which follows immediately. therefore, 2 line interface constitutes the pair of the system concerned -- all are contained between pipeline stages. namely, -- what is called -- others -- a device is contained, and when data is transmitted between this seed device and pipeline, 2 line interface between a precedence device and the 1st stage and between a succession device and the last stage is included. Namely, ACCEPT and VALID have a value of HIGH and LOW. [a signal] These values are respectively written as "H" and "L." When realizing this invention, a pipeline's most general application is usually digital. when it realizes as this kind of digital, a HIGH value is logical, for example -- it is "1" and a LOW value is logical -- even if it is "0", it does not interfere. However, when it is not necessarily restricted so that it may realize as digital, and it realizes as an analog, a system a HIGH value, It does not interfere, even if it is the quantity more than a certain voltage or other similar setting-out thresholds, and even if a LOW value is shown by the corresponding signal below the threshold of the same or others (or above), it does not interfere. In the case of digital application, for example, this invention can be realized using all known art like CMOS and bipolar **.
[0112]in order to provide the memory storage of a VALID signal, it is not necessary to use individual memory right [that] ** and wire In a digital example, this is truth. They are all for which to memorize directions of "validity" of data with data is only needed. If it only mentions as an example, in the digital television picture expressed with a digital value, a certain specific value is not permitted as specified to international standard CCIR601. In this system, since the sample of a picture is expressed, 8 bit binary numbers are used, and don't use zero and the value of 255.
[0113]When this kind of picture needs to be processed in the pipeline included in embodiment of this invention, one of the values of these (for example, zero) will be used in order to show that the data in the specific stage in a pipeline is not effective. Therefore, it should be considered that the data which is not all zero is effective. In this example, there is no specific latch who can say it is identifiable and is moreover memorizing "validity" of related data. In spite of it, the validity of data is memorized with data.
[0114]As shown in drawing 1, on an upper rightward arrow, the state of the VALID signal to each stage is shown as "H" or "L." Therefore, the VALID signal from the stage A to the stage B is LOW, and the VALID signal to the stage E is HIGH from the stage D. The state of the ACCEPT signal to each stage is shown as "H" or "L" on a

lower leftward arrow. Therefore, the ACCEPT signal from the stage E to the stage D is HIGH, and, on the other hand, the ACCEPT signal from the device connected downstream from the pipeline to the stage F is LOW.
[0115]When the ACCEPT signal which goes to a downstream stage from an upstream contiguity stage is HIGH, data is transmitted from one stage to another stage during 1 cycle period always (it explains below). When the ACCEPT signal between two stages is LOW, data is not transmitted among these stages.
[0116]When shade is again attached to one box with reference to drawing 1, a corresponding pipeline stage is assumed to contain effective output data, for example.Similarly, the VALID signal supplied from the stage concerned to the following stage is HIGH. A pipeline in case the stage B, D, and E contains effective data is shown in drawing 1. The stage A, C, and F does not contain effective data. In the starting point, the VALID signal to the pipeline stage A is HIGH, and it means that the data on the transmission line to a pipeline is effective.
[0117]unless similarly the ACCEPT signal to the pipeline stage F is LOW, therefore effective at this time in it being effective, it is alike, and is not involved and data is not transmitted from the stage F. Note being transmitted between [effective and] pipeline stages in invalid data. Overwrite of the invalid data which keeps valueless data is carried out, and, thereby, it removes invalid data from a pipeline. Valid data However, the object for processing or a pipeline stage, Since it must be kept in order to use it in the device or the device of the lower stream like a system connected to the pipeline who receives the data from the pipeline concerned, overwrite of the valid data must not be carried out.
[0118]in the pipeline who shows drawing 1, as for the stage E, the stage D the stage B including the effective data D2 including the effective data D1, including the effective data D3, The device (not shown) connected with the high pipeline is transmitted to the pipeline concerned, and contains the data D4 which should be processed. The VALID signal to the device which the stage B, D, and E follows from these stages or devices, respectively in addition to an upper device including [therefore] effective data is HIGH. However, since these stages do not contain effective data, the VALID signal from the stage A, C, and F is LOW.
[0119]The device downstream connected from the pipeline assumes data from a pipeline to be that in which carrier introduction ****** is not ready. A device transmits this state to the stage F by setting a corresponding ACCEPT signal to LOW. However, the stage F itself can receive data from the precedence stage E, excluding [therefore] effective data. Therefore, the ACCEPT signal to the stage E is set to HIGH from the stage F.
[0120]Similarly, as for the stage E, the preparation in which the stage F receives this data is complete including effective data. Therefore, as long as the effective data D1 is first transmitted to the stage F, the stage E can receive new data. If it puts in

another way, the stage F cannot transmit data downstream, but other stages of all the can be transmitted downstream, without writing all effective data in a pile, or losing it. Therefore, at the terminal point of the cycle 1, what only the step of 1 "is shifted for" to the right is possible for data. This state is shown in the cycle 2.
[0121]In the example shown in a figure, since the preparation which receives data with a downstream device new in the cycle 2 is not complete yet, the ACCEPT signal to the stage F is still LOW. Therefore, the stage F cannot receive new data. This is because overwrite of the effective data D1 will be carried out and it will be lost, if it does so. Therefore, since the stage E contains the effective data D2 similarly, the ACCEPT signal from the stage F to the stage E is set to LOW from the stage E like the case of the ACCEPT signal to the stage D. however -- all the stages from A to D can receive new data (or [that the stage of these does not contain effective data] -- or) Since these data can shift those effective data downstream and new data can be received, and these stages, This condition is transmitted to the contiguity stage which precedes just before by setting an ACCEPT signal corresponding, respectively to HIGH.
[0122]The condition of the pipeline after the cycle 2 in the period by which the line classification was carried out with the cycle 3 is shown in drawing 1. The preparation in which a downstream device receives new data from the stage F as an example assumes that it is not ready yet (the ACCPT signal to the stage F is LOW). Therefore, the stages E and F and overwrite of the invalid data whose stage D is effective ending with data D3 receipt and which was in this stage from before in the cycle 3 although still "having been blocked" are completed. Since the stage D cannot supply the data D3 in the cycle 3, this stage cannot receive new data, therefore sets the ACCEPT signal to the stage C to LOW. However, the preparation which receives new data is complete and stage A-C transmits this by setting a corresponding ACCEPT signal to HIGH. Note that the data D4 will be shifted from the stage A to the stage B.
[0123]Here, suppose that it is assumed that the preparation in which a downstream device receives new data in the cycle 4 will be completed. This downstream device transmits this to a pipeline by setting the FACCEPT signal to a stage to HIGH. Although stage C-F contains effective data, these stages are in this state and can receive new data so that it is [therefore] possible to shift data downstream. Therefore, since each stage can carry out the step of the data downstream only one step, these stages set an ACCEPT signal corresponding, respectively out of HIGH.
[0124]As long as the ACCEPT signal to the last pipeline stage (this example the stage F) is HIGH, the pipeline who shows drawing 1 operates as a stiffness pipeline, and only one step only shifts data for every cycle. Therefore, the data D1 contained in the stage F in the cycle 4 is shifted from a pipeline to the following device in the cycle 5, and other data of all the is shifted downstream only one step.
[0125]Here, it is assumed that the ACCEPT signal to the stage F is set to LOW in the

cycle 5. If this is explained again, data with new stage D-F cannot be received, and the ACCEPT signal to the last precedence contiguity stage will be set to LOW from these stages. Therefore, although the data D2, D3, and D4 cannot be shifted downstream, the data D5 is possible. Therefore, the state where the pipeline after the cycle 5 corresponds is shown in drawing 1 as the cycle 6.

[0126]If the desirable example of this invention is followed, the capability of the pipeline who does "fill up (fullness)" of the empty processing stage is highly advantageous. This is because a pipeline's processing stage is mutually released from combination with this capability. If it puts in another way, even if it is a case where the preparation which receives the data of a pipeline stage is not complete, it is not necessary to stop or and the whole pipeline does not need to wait for delay stages. When data not only with it but one effective stage cannot be received, the stage concerned only forms time "wall" in a pipeline. Even if it will be in this state, the stage of the lower stream of a "wall" can continue advancing data effective in the circuit connected to the pipeline, and can receive data with a still effective stage on the left-hand side of a "wall" so that RUKOTOSAE is possible, and can transmit it toward the lower stream. When some pipeline stages cannot receive new data temporarily, other stages can continue a normal system operation. Especially the pipeline can continue receiving data in the first stage A, unless the effective data which cannot be advanced since the preparation which receives data with the following new stage is not complete is contained in the stage A yet. As shown in this example, also in one or more processing stages or the state where it was blocked, data can be transmitted between a pipeline and a stage.

[0127]In the example shown in drawing 1, various pipeline stages assume that the ACCEPT signal received from the stage which adjoins immediately after them is not memorized. Instead, if the ACCEPT signal to a downstream stage is set to LOW, a flow will be traced back and spread by this LOW signal to the nearest pipeline stage that does not contain effective data. For example, with reference to drawing 1, the ACCEPT signal to the stage F assumes that it is what is set to LOW in the cycle 1. In the cycle 2, a LOW signal is returned and spread on the stage D from the stage F.

[0128]In the cycle 3, when the data D3 is latched to the stage D, an ACCEPT signal spreads four stages to the stage C in the direction which traces back a flow. When the ACCEPT signal to the stage F is set to HIGH in the cycle 4, this signal is spread to the stage C against a flow. If it puts in another way, change of an ACCEPT signal must be back spread through four stages. However, in the example shown in drawing 1, when there is an intermediate stage in which new data is acceptable, the ACCEPT signal does not need to spread all the courses back to a pipeline's starting point.

[0129]In the example shown in drawing 1, each pipeline stage still needs an individual input and an output data latch, in order to enable transmission of data between stages, without carrying out overwrite which is not intentional. the pipeline who shows drawing

1 -- "compression", although it is possible, Since a pipeline provides the stage which does not contain the effective data between the stages containing effective data when a downstream pipeline stage is blocked (i.e., when these stages cannot supply the data which these stages contain), "it does not expand." It depends for the capability compressed not right [that] on the cycle equivalent to the period in which data effective in the 1st pipeline stage does not carry out the whereabouts.
[0130]for example, the ACCEPT signal to the stage F continuing being LOW, and in the cycle 4, When the pipeline stage A and 3 are filled by effective data, as long as effective data continues carrying out the whereabouts to the stage A, for a pipeline, effective input data may be lost uncompressible more. In spite of it, the pipeline who shows drawing 1 reduces a risk of data being lost. As long as a reason has a pipeline stage which does not contain effective data, the pipeline concerned is based on the ability to compress.
[0131]In the logical method, both compression and expansion are possible, and other examples of the pipeline who has a circuit which restricts propagation of an ACCEPT signal to the nearest precedence stage ** are shown in drawing 2 and drawing 3. Although explained and illustrated in detail below about the circuit for realizing this example, drawing 2 and drawing 3 are useful in order to illustrate the working principle of this example.
[0132]In order to compare easily, the input data and the ACCEPT signal to the pipeline example shown in drawing 2 and drawing 3 are the same as the case of the pipeline example shown in drawing 1. Therefore, the stage E, D, and B has the effective data D1, D2, and D3, respectively. The ACCEPT signal to the stage F is LOW, and the data D4 is supplied to the start pipeline stage A. Three lines which connect the stage which became a pair which the pipeline stage of the couple of each neighborhood each adjoins are shown in drawing 2 and drawing 3. The line of the uppermost position which does not interfere even if it is a bus is a data line. A VALID signal is transmitted through this, a central line is a line, and the lowest line is a line to which an ACCEPT signal is transmitted through this. The ACCEPT signal to the stage F is [be / it / under / cycle 4 / removing] still LOW like a former case. In the cycle 4, the additional data D5 is supplied to a pipeline.
[0133]In drawing 2 and drawing 3, each pipeline stage is expressed as one block divided into two half portions, in order to illustrate that each stage in this pipeline's example contains primary and the 2nd data storage element. In drawing 2 and drawing 3, primary data memory storage is shown as a right half of each stage. However, only explanation should make this description the purpose and please understand that it is not what meant being restricted in this way.
[0134]As shown in drawing 2 and drawing 3, as long as the ACCEPT signal to a stage is HIGH, data is transmitted to the secondary memory element of the following stage from the primary storage element of the stage concerned during the arbitrary

prescribed cycle. Therefore, although the ACCEPT signal to the stage F is LOW, all -- others -- the ACCEPT signal to a stage is HIGH, and, as a result, the data D1, D2, and D3 are ahead shifted only one stage in the cycle 2, and the data D4 is shifted to the 1st stage A.

[0135]To this point, the pipeline example shown in drawing 2 and drawing 3 operates in the same method as the pipeline example shown in drawing 1. However, even if the ACCEPT signal to the stage F is LOW, the ACCEPT signal of ** is HIGH from the stage F to the stage E. Since there is a secondary memory element so that it may explain below, it is not required to spread to the upper stream for a LOW ACCEPT signal exceeding the stage F. A stage F signal transmits that the preparation which receives new data is complete by leaving the ACCEPT signal to the stage E with HIGH. The stage F is that (the ACCEPT signal to the stage F is LOW) which cannot transmit the data D1 in the primary storage element downstream in the cycle 3, therefore the stage E transmits the data D2 to the secondary-storage element of the stage F. Since primary [of the stage F] and secondary both the memory storage element contain the effective data which cannot be supplied in this state, the ACCEPT signal to the stage E is set to LOW from the stage F. Therefore, this means that a LOW ACCEPT signal merely spreads only one stage back to the cycle 2, and the complete process cycle to the stage C in the example shown in drawing 1 must be back spread by this ACCEPT signal in this case. Since stage A-E can supply those data, the ACCEPT signal from the stage concerned to a contiguity-just before them stage is set to HIGH. Therefore, the data D3 and D4 is shifted to the right only one stage. As a result, in the cycle 4, data is loaded to the primary data memory element of the stage E and the stage C, respectively. In this stage, although the effective data D3 in that primary storage element is included, the stage E can still use that secondary memory element, in order to memorize other data without a risk of carrying out overwrite of all the effective data.

[0136]In here, the ACCEPT signal to the stage F assumes that it is set to HIGH in the cycle 4 like a former case. This shows that the preparation in which the device of the lower stream to which a pipeline supplies data receives data from a pipeline is complete. However, since the ACCEPT signal of the stage F is set to LOW, it is shown in E stage that preparation in which the stage F receives new data has not been performed. or data is supplied [what as for the ACCEPT signal in each cycle period, in **, occurs in the following cycle, or] (ACCEPT HIGH) -- how -- or [or / that data must remain in a predetermined place] (ACCEPT LOW) -- please consider that it is shown how it is. Therefore, although the data D1 is supplied from the stage F even to the following device to from the cycle 4 before the cycle 5 and the data D2 is shifted to it from the secondary memory of the stage F to a primary storage, the data D3 in the stage E is not transmitted to the stage F. Since the ACCEPT signal of the following stage is HIGH, it is possible to usually pass along the data D4 and D5, and to

be transmitted to the next pipeline stage.
[0137]By comparing a pipeline's condition in the cycle 4 and the cycle 5, By equipping a secondary memory element shows releasing that expansion of the pipeline example shown in drawing 2 and drawing 3 is attained, i.e., a data storage element, so that the effective data in it can move forward. For example, since the data of these data blocks cannot be transmitted until the ACCEPT signal to the stage F is set to HIGH, in the cycle 4, the data block D1, D2, and D3 form "a solid wall." however, it being shifted from a pipeline, and the data D2 being shifted into the primary storage element of the stage F, and the data D1, once this signal is set to HIGH, The following device cannot receive the data D2, but when a pipeline has to do ****** "compression", the secondary memory element of the stage F will be in the state where it becomes free and new data can be received. This is shown in the cycle 6. During this cycle, the data D3 is shifted into the secondary memory element of the stage F, and usually passes along the data D4, and is supplied to the stage E from the stage D.
[0138]On the whole, drawing 4, drawing 5, drawing 6, and drawing 7 show a pipeline's desirable example. This desirable example realizes structure which a phase shows to drawing 2 and drawing 3 using 2 **** duplication clock of phi0 and phi1. Although 2 phase clock is preferred, please understand that it is also possible to drive various examples of this invention using the clock of two or more phases.
[0139]As shown in drawing 4, drawing 5, drawing 6, and drawing 7, each pipeline stage is illustrated as a thing with two individual boxes in which primary and a secondary memory element are shown. Like a former case, although a VALID signal and a data line connect various pipeline stages, in order to explain easily, they show drawing 4, drawing 5, drawing 6, and drawing 7 only an ACCEPT signal. As for change of the state in one clock phase of a certain ACCEPT signal, the change to HICH from LOW is shown in drawing 4, drawing 5, drawing 6, and drawing 7 using an upward arrow. Similarly, the change to LOW from HIGH is shown by the downward arrow. Memory element one of another data transfer from one memory element is shown by the open large arrow. The VALID signal from primary [of arbitrary predetermined stages] or a secondary-storage element is assumed to be certainly HIGH when a memory element contains effective data.
[0140]In drawing 4, drawing 5, drawing 6, and drawing 7, each cycle is shown as what comprises one perimeter term of the non-overlapping clock phases phi0 and phi1. So that it may be explained still in detail below data, In [during the clock cycle phi 1, it is transmitted to a primary storage element (shown as a box of the right in each stage) from a secondary-storage element (shown as a box of the left in each stage), and] another side, data -- during the period of the clock cycle phi 0 -- SONO from the secondary-storage element of one stage -- it is transmitted to the primary storage element of the following stage. Similarly, being further connected through an internal acceptance line so that an ACCEPT signal may be supplied in the same method, if

primary and a secondary-storage element is supplied [an ACCEPT signal] to a stage from a stage is shown. [in / in drawing 4 drawing 5, drawing 6, and drawing 7 / each stage] Thus, the secondary-storage element should know when the data can be supplied to a primary storage element.

[0141]Drawing 4, drawing 5, drawing 6, and drawing 7 show 1phicycle 1 phase. The data D1 already shifted to the secondary-storage element of the stage E, D, and B into this psych, respectively, D2, and D3 are shifted to the primary storage element of each stage. Therefore, the configuration of the pipeline in phi null effect phase of the cycle 1 is the same as the case of the cycle 1 of drawing 2. The ACCEPT signal to the stage F is assumed to be LOW like before. However, as shown in drawing 4, drawing 5, drawing 6, and drawing 7, the ACCEPT signal to the primary storage element of the stage F is LOW, but since this memory element does not contain effective data, this memory element sets an ACCEPT signal in that secondary-storage element at HIGH.

[0142]Since the secondary-storage element of the stage F does not contain effective data, an ACCEPT signal is set to HIGH as well as the inside from a secondary-storage element to the primary storage element of the stage E of the stage F. the data in all the upper primary one and secondary-storage elements since the primary storage element of the stage F can receive data like before -- all data -- overwrite ******** -- it can shift without things downstream. The shift of the data from one stage to the following stage is performed during the next phi null effect phase in the cycle 2. For example, ** -- the effective data D1 contained in the primary storage element of the stage E is shifted to the secondary-storage element of the stage F, and the data D4 is shifted into a pipeline, i.e., the secondary-storage element of the stage A.

[0143]The ACCEPT signal from a primary storage element to the secondary-storage element of the stage F is still HIGH, excluding [therefore] yet the data during the phi null effect phase in the cycle 2 with an effective primary storage element of the stage F. Therefore, data can shift only further 1 step during the phi null effect phase in the cycle 2 from the secondary-storage element to the right, i.e., the inside of each stage, to a primary storage element. However, effective data is once loaded to the primary storage element of the stage F, and when ACCEPT to the stage F from a downstream device is still LOW. Data cannot be shifted from the secondary-storage element of the stage F, without writing the effective data D1 in a pile, and destroying. Therefore, the ACCEPT signal from a primary storage element to the secondary-storage element of the stage F is set to LOW. However, excluding data with an effective secondary memory, since the ACCEPT signal output was HIGH, the data D2 can still be shifted to secondary ****** of the stage F.

[0144]Although data can be shifted into all the stages before it during phi1 phase period of the cycle 3, it cannot shift the data D2 into the primary storage element of the stage F. However, once effective data is loaded in a secondary-storage element, the stage F cannot supply this data. The stage F transmits this event by setting that

ACCEPT signal out of LOW.
[0145]When the ACCEPT signal to the stage F assumes [that it continues being LOW and], the data of the upper stream of F stage, It is possible to be shifted between stages and into the stage of each clock phase until the effective following data block D3 reaches the primary storage element of the stage E. As shown in a figure, this state is reached during the phi1 phase of the cycle 4.
[0146]Loading to the primary storage element of the stage E of the data D3 is completed during the phi phase of the cycle 5. Since this data cannot be shifted further, the ACCEPT signal from the primary storage element of the stage E is set to LOW. Upper data can be shifted normally.
[0147]here, as shown in the cycle 5 of drawing 2, suppose that the device boiled and connected downstream from the pipeline assumes that pipeline data is acceptable. This device transmits this event by setting an ACCEPT signal in the pipeline stage F during the phi1 phase of the cycle 4 at HIGH. In here, the primary storage element of the stage F shifts data to the right, and the primary storage element of the stage F can receive new data similarly. Therefore, since the data D1 was shifted during the phi1 phase of the cycle 5, the primary storage element of the stage F has not already contained the data which must be saved. Therefore, the data D2 is shifted from a secondary-storage element to a primary storage element into the stage F during the phi1 phase of the cycle 5. The secondary-storage element of the stage F can accept new data and signal by setting an ACCEPT signal in the primary storage element of the stage E at HIGH. in one stage during the data transfer from secondary [the] to the primary storage element. Although the memory element of both groups contains the same data, since the data in a secondary-storage element is held similarly in a primary storage element, overwrite is possible without a data loss for this data in a secondary-storage element. It is truth similarly about the data transfer from a primary storage element to the secondary-storage element of the following stage of one stage.
[0148]In here, the ACCEPT signal to the primary storage element of its stage F assumes that it is what is set to LOW during the phi plane 1 of the cycle 5. This means that the stage F cannot ****** the data D2 from a pipeline. Therefore, the stage F sets the ACCEPT signal from the secondary-storage element to the temporary storage element to LOW, in order to prevent overwrite of the effective data D2. However, overwrite cannot be carried out without a loss by the data D2 memorized by the secondary-storage element of the stage F, and, as a result, the data D3 is transmitted in the secondary-storage element of the stage F during the phi0 phase of the cycle 6. The data D4 and D5 cannot be downstream shifted as an all seems well. Once the effective data D3 is memorized by the stage F with the data D2, As long as the ACCEPT signal to the primary storage element of the stage F is LOW, not all secondary-storage elements can receive new data, and a secondary-storage element transmits this by setting the ACCEPT signal to the stage E to LOW.

[0149]The ACCEPT signal to a pipeline from a downstream device to HIGH, from LOW. Or when [that] changing conversely, don't spread this change to the upper stream further from the memory element which precedes just before in a pipeline (or inside of the same stage and a precedence pipeline stage). Only one memory element block per clock phase spreads this change in the upper stream in the pipeline concerned.
[0150]As explained to this example, the concept of "one stage" in the pipeline structure shown in drawing 4, drawing 5, drawing 6, and drawing 7 is a problem in connection with consciousness to some extent. When data carries out the whereabouts between stages, the data concerned, Since it is transmitted and is carried out into a stage (to secondary-storage element of the primary storage element of an upper stage to an adjoining downstream stage) (from a secondary-storage element to a primary storage element), One stage thinks that the "primary" memory element which follows by a "secondary-storage element" is comprised instead of [in the case of being shown in drawing 4, drawing 5, drawing 6, and drawing 7], and does not interfere. Therefore, most concepts of ["primary"] and a "secondary" memory element are the problems of labeling. In drawing 4, drawing 5, drawing 6, and drawing 7, when data is transmitted to the following stage or device from one stage, a primary storage element is an element by which data is taken out from it.
And since a "secondary" memory element is an "input" memory element and is obtained to the same stage, a "primary" memory element can also be referred to as an "output" memory element.

[0151]As shown in drawing 1 and drawing 3, when the above-mentioned example was described, reference was made only about transmission of the data under control of ACCEPT and a VALID signal. Before supplying between the internal memory elements of each pipeline stage, or before supplying the next pipeline stage, each pipeline stage should understand that it is possible to process similarly the data which it received arbitrarily. Therefore, also once with reference to drawing 4, drawing 5, drawing 6, and drawing 7, a pipeline stage can be defined as some pipelines who process arbitrarily the data memorized by the memory element including an input and an output memory element.
[0152]From the pipeline stage F, the downstream "device" does not need to be the hardware structure of a certain kind of other types, and rather, even if it is other pipeline itself or its portion, it does not interfere. So that it may explain below a pipeline stage, In order that a stage may end processing of the data, not only when all the downstream memory elements are filled with effective data, but when you need two or more clock phases, it can set the ACCEPT signal to LOW. As for this, a pipeline stage may happen similarly, one of memory elements, or when both making the effective data inside. If it puts in another way, it is not required for a downstream

memory element to only supply an ACCEPT signal for a stage based on whether the effective data which cannot be supplied is included immediately. Rather, into the stage concerned, the ACCEPT signal itself can be changed by an external circuit for the stage concerned, in order to control the passage for data between contiguity memory elements. In a similar method, a VALID signal can be processed similarly.

[0153]The big advantage of 2 line interface (it is one wire to each of VALID and an ACCEPT signal, respectively) is having without a control signal required in order to spread all the courses for a pipeline toward back to the start stage the capability controlling a pipeline. ** "tells" that the stage F cannot receive data to the stage E tells the stage E to the stage D by supposing that the cycle 3 of drawing 1 is referred to again, and the stage D is told to the stage C. When there are many stages rather than the effective data contained as a practical question, this signal may spread back to a distance further along with a pipeline. In the example and the cycle 3 which are shown in drawing 4, drawing 5, drawing 6, and drawing 7, from the stage E, no LOW ACCEPT signal is spread in the upper stream, but remains in it by the primary storage element.

[0154]This example can attain this flexibility, without attaching to especially a silicon part required in order to realize the design concerned so that it may explain below. Generally, each latch in the pipeline used for data storage needs only one transistor too much (silicon is suited very efficiently). In order to process ACCEPT and the VALID signal relevant to a data latch in each half-stage, it is preferred to add the gate of two latches and a small number too much.

[0155]The hardware structure which realizes a stage as shown in drawing 4, drawing 5, drawing 6, and drawing 7 is shown in drawing 8.

[0156]Although it is only an example for explaining, it is assumed that 8 bit data are transmitted in parallel through a pipeline (without performing operation beyond it in the combination logic circuit as an option, or carrying out). However, in order to realize this invention, please understand that either 1 bit data or the data below 8 bits can be used. 2 line interface according to this example is suitable for using it with the data bus of all width.

And the width of a data bus may differ for every stage according to the necessity for specific application.

The interface according to this example can be used also in order to process an analog signal.

[0157]As already inquired, it is usable in the conventional timing arrangement different from this, but as for an interface, being controlled by 2 **** duplication clock is preferred. In drawing 8 - drawing 16, these clock phase signals are referred to as PH0 and PH1. In drawing 8, one line is shown in each clock phase signals.

[0158]Input data is inputted into a pipeline stage via multibit data bus IN data, and is transmitted to the next pipeline stage or its next receiving circuit via output data bus

OUT data. Input data is loaded to a series of input latches (it is one so much to each input data signal) which are collectively referred to as LDIN first and constitute the above-mentioned secondary-storage element by the method shown below.
[0159]In the example of this example shown in a figure, all the latches' Q output follows up those D inputs, namely, when clocked into is HIGH, all the latches' Q output is loaded in a logic "1" level. Q output holds the value of those last. If it puts in another way, Q output "is latched" by descent **** of those corresponding clock signals.
[0160]Each latch has the Osamu AND combination of 1 of one or two non-overlapping clock signal PH0, PH1 (it is shown in drawing 9 and drawing 10 like) or these clock signal PH0, and PH1, and one logic signal to the clock. However, this invention operates good to the same extent by providing the latch who does a latch operation at the rise end of a clock signal, or other known arrangement for a latch, as long as the conventional method is applied, in order to guarantee the suitable timing of latch operation.
[0161]The output data from the input data latch LDIN, Supply through arbitrary and the combination logic circuit B1 as an option, and this combination logic circuit, Since it is provided in order to change the output data from input latch LDIN into intermediate data next, this intermediate data is loaded to output data latch LDOUT later, and this output data latch comprises the above-mentioned primary storage element.
[0162]Before supplying even the following device to a downstream direction as OUT data, the output from output data latch LDOUT is arbitrary, and can be similarly supplied through combination logic circuit B-2 as an option. It does not interfere, even if the following device is other pipeline stages or other devices which were connected to the pipeline.
[0163]In realization of this invention, each stage of a pipeline contains reasonableness test input latch LVIN, reasonableness test output latch LVOUT, acceptance input latch LAIN, and acceptance output latch LAOUT similarly. As for these four latches' each, it is preferred that he is an easy single stage latch. The outputs from latch U/IN, LVOUT, LAIN, and LAOUT are QVIN, QVOUT, QAIN, and QAOUT, respectively. The output signal QVIN from a reasonableness test input latch is connected to reasonableness test output latch LVOUT through the circuit which changes a direct or middle logical device or a signal as an input.
[0164]Similarly, the output reasonableness test signal QVOUT of a predetermined stage does not interfere, even if it connects with the input of reasonableness test input latch QVIN of the following stage through a direct or middle device or a logic circuit. Even if the device in this case is a device which changes a reasonableness test signal, it does not interfere. This output QVIN is similarly connected to a logic gate (it explains below), and the output of this logic gate is connected to the input of

acceptance input latch LAIN. The output QAOUT from acceptance output latch LAOUT is arbitrarily connected to a similar logic gate (it explains below) through other logic gates.
[0165]As shown in drawing 8, the output reasonableness test signal QVOUT directs data effective in the following circuit which forms OUT VALID receivable as an IN VALID signal by the stage which follows, or is only connected to a pipeline. Data acceptance preparation of that next circuit or a stage is directed on each stage as the signal OUTACCEPT, and this signal is connected to acceptance output latch LAOUT as an input through the logic circuit explained below preferably. Similarly, the output QAUT of acceptance output latch LAOUT is connected to acceptance input latch LAIN as an input through the logic circuit explained below preferably.
[0166]When realizing this invention, the output signals QVIN and QVOUT from the reasonableness test latch LVIN are combined with the acceptance signal QAOUT and OUT ACCEPT1, respectively, in order to form the input to the acceptance latches LAIU and LAQUT. In the example shown in drawing 8, these input signals presuppose that it is logically [each acceptance output signal QACUT and OUT ACCEPT] reverse, and are formed as a logic NAND combination of each reasonableness test signal QVIN and QVOUT. The conventional logic gates NAND1 and UAND2 carry out NAND operation, and inverter INV1 and INV2 form the logical reverse of each acceptance signal. When a NAND gate is arbitrary or all the input signals are in a logic "0" state in digital design art so that it may be common knowledge, the output from a NAND gate is logic "1." Therefore, it restricts, when a logic "1" state has all the inputs of a NAND gate, and the output from a NAND gate is logic "0." As it is common knowledge similarly in the technical field concerned, the output of a digital inverter like INV1 is logic "1" when the input signal is "0", and when the input signal is "1", it is "0."
[0167]Therefore, when "NOT" shows binary reversal, the inputs of NAND gate NAND1 are QVIN and NOT (QACUT). The input to the acceptance latch LAIN can be decomposed as follows using known art.
[0168]or [that the signal QVIN will be "0" if a NAND(QVIN, NOT (QAOUT)) =NOT(QVIN) OR QAQUT paraphrase is carried out] -- or, The signal QAQUT is "1" or, in both cases, the combination of inverter INV1 and NAND gate NAND1 is logic "1." Therefore, one of inputs can accept gate NAND1 and inverter INV1, and they can be coupled directly with the latch's LAOUT QAOUT output, and one input can realize them now by one single OR gate combined with reversal of the output signal QVIN of reasonableness test input latch LVIN.
[0169]Similarly, in digital design art, many latches suitable for the use as a reasonableness test and an acceptance latch can have the two outputs Q and NOT (Q), i.e., Q, and logical reverse of those so that it may be common knowledge. Therefore, when this kind of latch is selected, one input to OP gate (therefore) can be

coupled directly with the reasonableness test latch's LVIN NOT (Q) output. Gate NAND1 and inverter INV1 are realizable using a well-known Prior art. However, the latch who does not reverse an output may be used according to latch's architecture used, and it may be still more efficient to instead provide gate NAND1 and inverter INV1. These both can be efficiently realized in a silicone device in a similar manner. Therefore, in order to generate a Q signal and/or its logical reverse, arbitrary known arrangement can be used.

[0170]When the output from both clock signals (PH0 in an output side and PH1 in an input side) and the acceptance latch of the same side is logic "1", data, the reasonableness test latches LDIN, LDOUT, and LVIN, and LVOUT load each of those data input. Therefore, the output of a clock signal (PH0 to input latch LDIN and LVIN) and each acceptance latch (LAIN in this case) is used for the method of logical AND, when they are [both sides] logic "1", it is restricted, and data is loaded.

[0171]for example, a latch CMOS realization -- the logical AND operation which controls latch's loading (pass CK shown in a figure, or operation-ization "input") in specific application [like], In the conventional method, it is easily realizable by connecting each operation-ized input signal (for example, PH0 for the latches LVIN and LDIN and QAIN) to the gate to the MOS transistor by which the series connection was carried out to the latch's input line. Therefore, it is required to provide a actual logic AND gate, and the problem of the timing which originates in the propagation delay in high-speed application by this may be caused. Therefore, the AND gate shown in a figure only directs the AND function which should be carried out in order to generate various latches' operation-ized signal.

[0172]Therefore, when PH0 and QAIN are "1", they restrict both sides, and the data latch LDIN loads input data. This data latch latches this data, when one of these two signals is set to "0."

[0173]In order to carry out the clock of the data and a reasonableness test latch of the input (and output) side of a pipeline stage, only one of the clock phase signal PH0 or PH1 is used, but. Another clock phase signal is directly used, in order to carry out the clock of the acceptance latch of the same side. As for one near (an input or an output) acceptance latch of the pipeline stages, if it puts in another way, it is preferred to carry out the clock of the "out of phase" using the data of the same side and a reasonableness test latch. For example, in order that PH0 may generate clock signal CK the data latch LDIN and for the reasonableness test latch LVIN, it is used, but PH1 is used in order KOROKKU [an acceptance input latch].

[0174]As a pipeline's example of an operation enlarged by 2 line reasonableness test and the acceptance circuit, the data with either a precedence pipeline stage or a transmission device effective in the beginning is assumed to be what is not provided in the input to the circuit concerned. Since the system concerned was extremely reset recently when putting in another way, the reasonableness test input signal IN VALID to the stage shown in a figure is assumed not to be "1" yet. Since the system was reset at the end, several times of clock cycles occur, and the circuit concerned is assumed to have reached regularly as the result. Therefore, the reasonableness test input signal QVIN from the reasonableness test latch LVIN is loaded as "0" during positive [next to / the / clock PH0]. Therefore, the input to acceptance input latch LAIN (passing gate NAND1 or other equivalent gates) is loaded as "1" during positive [next to / the / clock signal PH1]. That is, since the data in the data input latch LOIN is not effective, (since the stage concerned holds no data value storage), as for the stage concerned, the stage concerned transmits that acceptance preparation of input data is complete.

[0175]In this example, in order to enable the operation of data and the reasonableness test latches LOIN and LVIN, note that the signal IN ACCEPT is used. Since the signal IN ACCEPT is "1", at this time these latches, It acts effectively as a conventional penetration latch, and as the result, all the data on IN data bus is only loaded to the data latch LOIN, shortly after the clock signal PHD is set to "1." Of course, as long as the output QAOUT from that acceptance latch is "1", this invalid data is similarly loaded to that next data latch LDOUT of the pipeline stage which follows that next.

[0176]Therefore, unless the data latch contains effective data, "this latch loads" all the data provided among a cycle positive [next to / that / each clock signal]. [receive namely] On the other hand, such invalid data is not loaded to the stage of all of as [whose acceptance signal from a corresponding acceptance latch is a low (namely, "0") to it]. As long as the corresponding INVALID (or QVIN) signal to a reasonableness test latch is a low, the output signal (the reasonableness test input signal to the following reasonableness test latch is formed) from a reasonableness test latch is still "0."

[0177]This is directed when the input data to a data latch is effective, and the reasonableness test signal IN VALID goes up to "1." Next, corresponding reasonableness test latch's output rises to "1" in the next rise end of each clock phase signal. For example, in the next rise end of clock phase signal PH0, when a corresponding IN VALID signal becomes a high (that is, it goes up to "1"), the latch's LVIN reasonableness test input signal QVIN goes up to "1."

[0178]Instead, the data input latch LDIN assumes here that it is the thing containing effective data. When the preparation which receives the data output latch's LDOUT new data is complete, the acceptance signal QAOUT is "1." In this case, in the cycle of that next positive [of clock signal PH1], the data latch LDOUT and the reasonableness test latch VOLT are made operation possible, and the data latch LDOUT loads the data which carries out the whereabouts to that input. Since a clock signal is non-redundancy, the above-mentioned operation is generated before the rise end next to [the] another clock signal PH0. Therefore, in the next rise end of PH0, a precedence data latch (LDIN) is not latched in the new input data from a precedence

stage until the data output latch LOOUT finishes latching safely the data transmitted by the latch LDIN.
[0179]Therefore, since it operates based on the mutual phase of a clock, the same sequence is continued by all the nearest neighbors of the data latch (between the inside of a stage, or a contiguity stage) in which a receipt is possible in data. Since the effective data which cannot yet be supplied is included, all the data latches in which the preparation which receives new data is not complete, It has an output acceptance signal (QA output from the acceptance latch LA) which is LOW, and the data latches LDIN and LDOUT are not loaded. Therefore, as long as one near (an input or an output) acceptance signal (output from an acceptance latch) of a predetermined stage or a certain stage is LOW, the corresponding data latch is not loaded.
[0180]The reset function included in a desirable example is shown in drawing 8. Reset-signal NOTRESET0 is connected to the reversal reset input (a small round head shows reversal as usual) of reasonableness test output latch LVOUT in the example shown in this figure. As everyone knows, this means that coercion is exerted always so that the reasonableness test latch LVOUT may output ″0″, when reset-signal NOTRESET0 turns into ″0.″ One advantage of resetting a latch, when a reset signal becomes a low (set to ″0″) is resetting a **** latch to interception of transmission. In this case, effective transmission starts, and when a reset signal is set to HIGH, these latches are in an ″empty″ state or the reset state always. Therefore, reset-signal NOTRESET0 operates as a digital ″ON and OFF″ switch. Therefore, this signal must be a HIGH value in order to operation-ize a pipeline.
[0181]Note that it is not required to reset all the latches that hold effective data in a pipeline. As shown in drawing 8, reasonableness test input latch LVIN is not directly reset by reset-signal NOTRESET0, but is indirectly reset. Reset-signal NOTRESET0 assumes that it descends to ″0.″ If the input to acceptance output latch LAQUT (pass gate NAND1) is set to HIGH, regardless of the state of the before, the reasonableness test output signal QVOUT will descend to ″0″ in a similar manner. Similarly, the acceptance output signal QAOUT goes up to ″1.″ Next, this QAOUT value ″1″ is transmitted to the acceptance input latch input LAIN regardless of a reasonableness test input signal QVIN state as ″1.″ Next, the acceptance input signal QAI goes up to ″1″ in the next rising edge of N clock signal PH1. If the reasonableness test signal IN VALID assumes that it was correctly reset by ″0″, Shelf ** and when the reasonableness test latch LVIN is reset directly, and it carries out, in the next rising edge of clock signal PH0, the output from the reasonableness test latch LVIN is set to ″C.″
[0182]As this example shows, in order to reset all the reasonableness test latches, it is only required to reset a reasonableness test latch to only one side of each of the stage (a final stage is included). In fact, in many applications, it is not required to reset other the reasonableness test latches of all the. Namely, when it can guarantee that

reset-signal NOTRESET0 continues and carries out the whereabouts to the perfect period of phase PH0 of the both sides of a clock and PH1 longer than 1 cycle. "Automatic reset" (back propagation of a reset signal) occurs to the reasonableness test latch period in a precedence pipeline stage. When it continues as a practical question during the number of complete cycles of the phase of the both sides of the clock of the number of pipeline stages, and the same number which carries out the whereabouts at least and a reset signal is held, it is only required to reset the reasonableness test output latch in the last pipeline stage directly.
[0183]As for drawing 9 and drawing 10, two side by which the pipeline stage in the example shown in drawing 8 to an order that the relation between non-overlapping clock signal PH0 and PH1, the effect of a reset signal, maintenance of data, and a reasonableness test differ from an acceptance signal variously was illustrated shows transmission of the data to the meantime. In drawing 9 and the example shown in the timing diagram of drawing 10, the outputs from the data latches LDIN and LDOUT are assumed to be the logical block B1 which intervenes, and a thing supplied without operation beyond this by B-2. This shows as an example and does not have a restrictions meaning. All the combination logical structures should understand that it is possible to make it contain between the data latches of the continuous pipeline stage or between the input of one single pipeline stage and an output side. The value (for example, HEX data word "aa" or "04") actually illustrated about input data is only an example of an illustration similarly. As long as accommodation and a latch, or memory of the value of each bit or an input word is possible for a data latch or other storage devices, the width of an input data bus is arbitrary as mentioned above (and even if it is an analog, it does not interfere).
Desirable data structure - "Token"
In the sample application shown in drawing 8, since supply of input data is enabled via **, such as the combination logical block B1 and B-2, and there is no control circuit which excepts a stage, each stage processes all the input data. In order to provide large flexibility, this invention has the data structure that a "token" is used, in order to distribute data and control information through the whole system. Each token comprises the binary bit of a dividing [into one or more token word blocks] series. It is equivalent to one one of three types shown below, namely, a bit is an address bit (A), a data bit (D), or an extension bit (B). Data assumes that it is what is transmitted as a word via the 0-bit bus which has 1 bit-extension bit lines, without only having a restrictions meaning not necessarily as an example. The example of a 4-word token is shown below in order of transmission.
1st word Word of 2nd E A A A D D DD D Word of 3rd E D D D D D D D D Word of 4th E D D D D D D D D E D D D D D D D D extension bit E, Note being used as an addition (preferably) to each data word. Even if the length of an address field is variable, it does not interfere, and it is preferred to be transmitted immediately after a word [1st]

extension bit.
[0184]Therefore, a token comprises one or more words of the digital data in this invention (binary). Each of these words is preferably set in order in parallel, and is transmitted. However, this transfer method is not necessarily required, and serial data transfer is also possible for it using ** and known art. For example, in an animation parser, control information is transmitted in parallel and data is transmitted in series in this case.
[0185]At the time of a start, each token has preferably one address field (string of A **BITTO) which identifies the type of the data contained in a token so that it may illustrate as an example. In almost all applications, the portion of one single word or one word is enough to transmit the whole address field. However, if only it receives the whole address field and the logic circuit is included in the corresponding pipeline stage which can memorize a certain amount of expression of a partial address field with sufficient length to decode, above conditions, It is not necessarily required for this invention.
[0186]In order to transmit an address field, note not needing a dedicated line or a dedicated register. An address field should be careful of being transmitted using a data bit. When it does not have intention of being operation-ized by the specific address field so that it may explain below, when it can supply, a pipeline stage is not slowed down without delay of a stage along with a token.
[0187]The remaining data in the token following an address field is not restrained by use of a token. These D data bits can take any value, and it is attached to these bits and a meaning is not important here. That is, the meaning of data can be changed by where [in a system] the data concerned is arranged at the specific time, for example. If it does not interfere even if the number of the data bits D attached behind the address field is long as it is required and it is short, and tokens differ, the number of data word may change substantially.To a pipeline stage, an address field and an extension bit are used in order to carry a control signal. Since the number of the words (string of D bit) in a data field is optional and it does not interfere, the information carried in a data field can also change according to it. Therefore, the following explanation is turned to use of an address and an extension bit. In this invention, when many blocks in a circuit are collectively connected to comparatively easy composition, a token is an especially useful data structure. The simplest configuration is a pipeline of a processing step as one of them is shown to drawing 1, for example. However, use of a token is not restrained by only the use in pipeline structure.
[0188]Each box is again assumed to be a thing showing a perfect pipeline stage. In the pipeline of drawing 1, data flows into the right from the left of a figure. Data goes into a machine and is supplied to processing stage A. If data may be corrected, this stage may not be carried out and will supply data to the stage B. Since it is complicated

when corrected, the number of the data items which generally flow into arbitrary stages is not the same as the number of outflows. **, such as the stage B correcting data again and supplying it to the stage C. In this kind of design, flowing into a counter direction is impossible for data. Therefore, for example, the stage C cannot supply data to the stage A. These constraints are accepted thoroughly in many cases.
[0189]On the other hand, even if it is a case where direct continuation of between these two blocks is not carried out, that information can be transmitted to the stage C has the dramatically desirable stage A. Communication of the stages A and C is restricted when passing through the stage B. One advantage of a token is having the capability a token attaining this kind of communication. All the processing stages that do not recognize a token only supply the token which is not changed to the following block.
[0190]Since one extension bit is transmitted with the address and data field in each token according to this example, the processing stage can supply a token (it may be optional length), without needing no to have to decrypt that address. According to this example, all the tokens that contain in it the extension bit which is HIGH ("1") are followed by the succeeding word which is a part of same token. Similarly, this succeeding word also has an extension bit and shows whether this bit has another token word in a token. When a stage encounters a token [of as / whose extension bit of the is LOW (one "0")] word, it turns out that the token word is the last word of the token concerned. Therefore, it is assumed that the following word is the 1st word of a new token.
[0191]A token should be careful of what is understood that it can apply to the process element of a still more complicated configuration although especially the easy pipeline of a processing stage is useful. The example of a complicated process element is shown below.
[0192]In the case of this invention, in order to transmit the word of the last of a predetermined token by giving the extension bit set to "0", it is not necessary to use the state of an extension bit. One alternative to desirable implementation planning is moving an extension bit, as the 1st word of a token is shown instead of the word of the last of a token. This can be attained by converting numerals decryption hardware appropriately.
[0193]It is not the 1st word of a token and, as for the advantage at the time of using the extension bit of this invention, in order to transmit the last word, it is useful to correct the operation form of a block of a circuit according to whether the token has an extension bit in many cases. The example of the aforementioned point of an argument is a token which operation-izes the stage which processes the animation quantized value memorized in the quantization table (generally memory device). For example, it is a table including the 8-bit arbitrary binary integer of 64.
[0194]In order to load a new quantization table to a pipeline's quantizer stage, a

"QUANT and table" token is sent to a quantizer. In such a case, for example, a token comprises the token word of 65. The 1st word contains the code "QUANT and table", namely, makes a quantization table. The word of 64 which is an integer of a quantization table follows this.
[0195]When coding a video data, it is required occasionally to transmit this kind of quantization table. In order to attain this function, QUANT and a table token without an extended word can be sent to a quantizer stage. When this token is seen and you have noticed that the extension bit of that 1st word is LOW, the quantizer stage can create QUANT and the table token which reads that quantization table and contains the quantization table value of 64. Until the new termination of the token concerned is shown by the LOW extension bit equivalent to the value of the 64th quantization table, The extension bit of the 1st word (it was LOW) is changed so that the extension bit concerned may be HIGH and the token concerned may continue in the state of a HIGH extension bit. This advances by a general method through a system, and is coded in the flow of a bit.
[0196]It supposes that explanation is continued using this example, and one of loading a new quantization table to the memory device of itself, or reading that table according to whether the 1st word of QUANT and a table token is ending with a set about that extension bit, is possible for a quantizer.
[0197]using an extension bit -- the 1st token word in one token -- or which of the last token word is transmitted -- it depends for that selection on the system by which a pipeline is used in it. Both alternatives are possible according to this invention.
[0198]Another alternative of a desirable extension bit proposal is including a length count when starting a token. For example, when a token is very long, this kind of device is sometimes efficient. For example, the length of the general token in predetermined application is assumed to be 1000 words. When the extension bit proposal (it has each bit by which token word attachment was carried out) shown in a figure is used, since all the extension bits are included, the additional bit of 1000 is required for a token. However, 10 bits is slightly required in order to code the length of a token in binary form.
[0199]Therefore, although there is a use of a long token, according to experience, it has become clear that there are also many uses of a short token. Here, a desirable extension bit proposal is advantageous. the length of a token -- in being 1 slight word, in order to transmit this, only 1 bit is needed. However, the same 10 bits as the aforementioned case are needed for a count proposal.
[0200]The fault of a length count proposal is shown below. 1) It is inefficient to a short token. 2) The length of a token must be known before generating three (it is slightly countable at 10 bits if it is less than 1023 words) counts restrained in respect of the greatest length to a token (it is probably at the time of a token start). 4) All the circuit blocks treating a token need to be provided with the hardware for counting a word. 5)

It is not clear whether recovery can be attained or not, when a count rots (for a data transmission error).
[0201]The advantage of an extension bit proposal based on this invention is shown below. 1) Since an unrecognized token can be correctly supplied by considering only an extension bit, a pipeline stage does not need to be provided with the circuit block which decrypts all the tokens. 2) Coding of an extension bit is the same about all the tokens. 3) There is no limit about the length of a token. 4) This plan proposal is efficient to a short token (related with the overhead for expressing the length of a token). 5) Error recovery is attained automatically.Or [when an extension bit rots, / one random token is generated] ″(related with the extension bit decomposed from ″1″ to ″0″) or a token is lost (a corrosion extension bit is from ″0″ to ″1″). Only within a related token, the whereabouts of the problem is carried out locally. After the token concerned, right operation is resumed automatically.
[0202]It does not interfere, even if the length of an address field changes. Since it is possible to squeeze the most general token to a very small number of words by this, it is highly advantageous. Since all the processing stages guarantee that it can operate continuously with a total bandwidth, in a video data pipe-line system, this is dramatically important as a result.
[0203]In order to make the address field of variable length possible based on this invention, an address is selected so that the short address which random data follows may never be confused with a longer address. the desirable art (it serves also as a ″code″ for operation-izing the intended pipeline stage) for coding an address field is the art of the common knowledge which is first indicated by Huffman and is called a common name ″Huffman code.″ However, if it is the usual expert in the technical field concerned, saying that it will succeed even if it uses other coding proposals should be understood.
[0204]Although Huffman encoding is well understood in the field of a digital design, the following example is shown in order to provide a general background.
[0205]A Huffman code comprises the word constituted by the symbol string (for example, generally in the context of a digital system like this invention, a sign is a binary digit). The longer code word which starts with the sign which forms a code word with shorter character of those with ****** and the Huffman code word with the special length of a code word is that a code word is chosen so that it may not be possible. As for a token address field, if based on this invention, being selected using known Huffman encoding art is preferred (it is not necessarily required).
[0206]In this invention, it is preferred that an address field starts in the most significant bit (MSB) of the 1st word token. (The clear statement about MSB can correct the case so that various clear statement it is arbitrary and concerning MSB may be accommodated.) An address field is continued more via the low contiguity bit of importance. In specific application, when a token address needs two or more token

words, the address field which is a least significant bit within all predetermined words is continued in the most significant bit of the following word. The minimum length of an address field is 1 bit.
[0207]In order to generate the token used in this invention, arbitrary one out of several sorts of known hardware structures can be used. One of this kind of the structures is the state machine by which the micro program was carried out. However, even if it uses a known microprocessor or other devices, it does not interfere.
[0208]The main advantages of a token plan proposal based on this invention are having the adaptability to the necessity of not having been predicted. For example, as for this introduction, when a new token is introduced, it is common to influence only the pipeline stage of a **** small number. When most possible, it is that two stages or a circuit block is influenced slightly. in order that [namely,] one block may generate a token in the first place first and one block or stage may treat this new token now -- new -- a design -- or it is corrected. Be careful of them and make other pipeline stages not convert at all. These stages, not to mention it, can treat a new token, without converting the design of these stages, since the token which does not recognize this new token, therefore is not converted is supplied.
[0209]It is a clear advantage of this invention to have the capability to change the existing designed device into the state where it is not influenced substantially. By improving some semiconductor chips other than [said] in the aforementioned chip set for some semiconductor chips in one chip set in a design surface, it changes into the state where it is not influenced thoroughly, and Lycium chinense is possible in **. This is advantageous from the position of both a customer and a chip maker. Though reconstruction means that all the chips are influenced by a design variation, since the reuse of (the conditions that the progress level of integration becomes high gradually and the number of the chip in a system falls), and the same design can be carried out, In the point of the time taken to be marketed, whether it being said that the grade which can be attained is excelled, and the becoming advantage are left behind.
[0210]Be careful of the conditions which occur when it is necessary to extend a token set so that two word addresses may be included especially. In this case, skill is not yet required for converting the existing design, either. The token decoder in a pipeline stage tries to decrypt the 1st word of this seed token, and concludes that the 1st word does not recognize a token. Next, this token decoder supplies the token converted by using an extension bit, in order to carry out this operation correctly. This token decoder is that of "assuming" as the 2nd word is a part of data field of the token which this token decoder does not recognize, (though the token concerned contains an address bit) This token decoder does not try to decrypt the 2nd word of the token concerned.
[0211]In many cases, a pipeline stage or the connected circuit block converts a token. Generally, this is not as a necessary condition and takes the form where a KOKUN

data field is converted. It is common to change the number of the data word in the token which it is going to convert by the method of whether specific data word is removed and the paddle gap which adds new data word. Depending on the case, a token is thoroughly removed from the flow of a token.
[0212]generally in most applications, a pipeline stage decrypts the token of 2 and 3 -- it is (operation-ized by the token). That is, a stage is supplied without not recognizing other tokens and changing other tokens. In very many cases, only one token (i.e., data token ADO itself) is decrypted.
[0213]It depends for the operation of a special stage on the result of the operation of the past of the very thing in many applications. Therefore, it depends for the "state" of a stage on the state before that. If it puts in another way, stages will be other methods of saying that this must hold some information about the history [in / one or more than it / in the stage concerned / a front clock cycle] of itself depending on the memorized state information. This invention is well suitable in the use in the pipeline containing this kind of "state machine" stage, and the use in application which is a pipeline latch with an easy latch in a data path.
[0214]It is an important advantage of an invention that 2 line interface based on this invention has conformity to this kind "state machine" (state machine) circuit. This is truth especially when a data path is being controlled by a state machine. In this case, the above-mentioned 2 line interface technology can be used in order to guarantee that the "current state" of the machine doubled the data and the pace under control, and has stopped in a pipeline.
[0215]In order to decrypt a token address field, the lineblock diagram in which one example of the circuit included in a pipeline stage was simplified is shown in drawing 11. This figure shows a pipeline stage with the characteristic of a "state machine." Each word of a token will be HIGH if this bit has many words from this in the token concerned including one an "extension bit."
Or this is ** LOW which is the last word of a token.
When this is the last word of a token, the effective data word of the next is the start of a new token, therefore the address must be decrypted. Therefore, it depends for the determination about whether the token address in all predetermined words is decrypted on getting to know the value of a front extension bit.
[0216]Suppose that it supposes that 2 line interface is not illustrated (accepting and a reasonableness test signal and a latch) only for the purpose of making a drawing brief, and omits resetting a circuit about all the details to treat. Like before, 8 bit data word is not for restrictions, and is assumed as a mere example.
[0217]The pipeline stage as this example delays a data bit and an extension bit only one pipeline stage. This stage decrypts a data token. When the 1st word of a data token is supplied to a circuit output, the signal "data ADDR" is made and it is set to HIGH. A data bit is delayed by the latches LDIN and LDOUT, and these latches' each

is repeated 8 times to eight data bits used for this example (it corresponds to 8 input 8 output latch). Similarly, an extension bit is delayed by the extension bit latches LEIN and LEOUT.

[0218]In this example, it has the latch LEPREV in order [of an extension bit] to memorize the latest state most. The value of an extension bit is loaded to LEIN, and is loaded to LEOUT in the next rising edge of non-overlapping clock phase signal PH1. Therefore, although the latch LEOUT contains the value of the present extension bit, it is restricted only to throughout [second half / of non-overlapping 2 phase clock]. However, the latch LEPREV loads this extension bit value in that next rising edge of the identical signals whose operation [0 /, i.e. extension bit input latch LEIN, / clock signal PH] is enabled. Therefore, the latch's LEPPEV output QEPREV holds the value of an extension bit during the front PH0 clock phase. The result of having added the latch's LDIN noninverting MD [2] to 5 bits of the data word output from a reversal Q output is combined with the extension bit value QEPPEV of before in a series of logic gate NAND1, NAND2, and NOR1. It is [in / operations / such / the field of digital design art] common knowledge. Nominal "NMD [m]" shows logical reversal of the bit m of midday towered MD [7:0]. When known Boolean algebra art is used, it restricts, when a front extension bit is "0" (QPPEV= "0"), and output signal SA from this logical block (output from NOR1) is HIGH ("1"), ** may be shown and it turns out that the data word structure in the noninverting Q latch's (original input word) LDIN output is "000001xx." That is, all the MD[7]-MD [3] bits that are five high bits are "0", and bit MD [2] is "1", and the bit in a zero **WAN position has all any value.

[0219]Therefore, four possible data word sets to HIGH (there are four permutations of "xx"), SA, and the output of the address signal latch LADDR by whom input SA is being connected to the input if it pulls. If it puts in another way, when one of the four possible proper tokens will be supplied to this stage, And the data word of a front when a front extension bit is zero, It restricts, when it means that it is the last word in a series of front token words, and the present token word is the 1st token word in the present token, and an operation-ized signal (data ADDP "1") is supplied.

[0220]Therefore, when the signal QPREV from the latch LEPREV is LOW, the value in the latch's LDIN output is the 1st word of a new token. Gate NAND1, NAND2, and NOR1 decrypt a data token (000001xx). However, this address decoded signal SA is delayed in the latch LADOP so that signal data ADDR may have output data OUT data and the same timing as OUTEXTN.

[0221]Another, easy example of a state subordinate pipeline stage based on this invention is shown in drawing 12. This example generates the signal LASTOUTEXTN, in order to show the value of front output extension bit OUTEXTN. Respectively one of present and two-izing signals (it can set to CK input) to the last extension bit latches LEOUT and LEPPEV which can be operated. the (case with effective data where come out and it is being accepted [are and] -- an output reasonableness test

and Q output from the acceptance latches LVOUT and LAOUT are HIGH(s), respectively). It is derived from gate AND1 so that these latches may load only the new value to these latches. Thus, these signals hold only an effective extension bit and the value which is not the truth connected with the data which is not effective is not loaded to these signals. In the example shown in drawing 12, 2 line effective / acceptable logic includes OR1 and OR2 gate to which a downstream acceptance signal and the input signal which comprises each of reasonableness test latches' LVIN and LVOUT noninverting output are supplied. This shows one method for which replacement of gate NAND1/2 in drawing 8, and INV1/2 is possible, when a latch has an inverted output.
[0222]On the whole, it is truth that all the latches who restrict when data is actually transmitted between pipeline stages, and hold state information although this is the very easy example of a ″state subordination″ pipeline stage, i.e., the example depending on the state of only one single bit, are updated. If it puts in another way, data is both share effect, and it will be restricted when accepted by the following stage. Therefore, in order that this kind of latch may guarantee being reset appropriately, it is careful, and if it is ******, there is nothing.
[0223]It being based on this invention, and generating and using a token provides many advantages rather than the coding technology of the common knowledge for transmitting data via a pipeline.
[0224]As already explained to the 1st, in order to provide an efficient expression of a general token, even if the length of the address field of a token is variable, it does not interfere (and for example, Huffman encoding can be used).
[0225]By performing the coding which is [2nd] consistent about the length of a token, Even if it is a case where the token concerned is not recognized by the token decoder circuit in a predetermined pipeline stage, it makes it possible to process the end of a token correctly (therefore, the start of the following token) (being easy and non operation transmission are included).
[0226]Communication between one stage and the stage of the lower stream which is not the nearest contiguity stage in a pipeline is enabled according to the rule (namely, while not having been changed, in order to supply a token) and hardware structure for treating an unrecognized token to the 3rd. Since this enables change in the future of a token set, without needing large-scale redesign of the existing pipeline stage, it increases a pipeline's extensibility and efficient adaptability. The token of this invention is especially useful, when it is used with 2 line interface so that it may be explained to the above and the following.
[0227]The function of the pipeline stage as an example shown in drawing 13 and drawing 14 is explained below. When the token (a data token is called in this example) by which the stage was planned is processed, the stage concerned doubles all the words other than the 1st word including the address field of the data token in this

token. On the other hand, when a stage is processing the arbitrary tokens of other kinds, the stage concerned deletes all the words. As for an overall effect, only a data token appears in an output, and each word in these tokens is repeated twice.
[0228]Even if many parts of the system illustrated here are the same as the parts shown in drawing 4, drawing 5, drawing 6, and an easy structure for whether being Haruka who shows drawing 7, they do not interfere. This illustrates an important advantage. Since it converts slightly or the same 2 line interface can be used without all reconstruction, even if it is a still more complicated pipeline stage, it has the same advantage about flexibility and elasticity.
[0229]The data doubleness stage shown in drawing 13 and drawing 14 is only a mere example of the operation of a different type which can carry out a pipeline stage in predetermined application and which exists innumerably. However, this "doubleness stage" may form a "bottleneck" and the pipeline based on this example shows TEJI [like] "which serves as a pack together" as that result.
[0230]In order to carry out the operation, a "bottleneck" requires comparatively long time, or they may be all stages that make more data in a pipeline rather than having received. This example shows that it can be very easily adapted for the application with which 2 line acceptance differs from the effective interface based on this example.
[0231]The doubleness stage shown in drawing 13 and drawing 14 has two latches LEIN and LEOUT who latch the state of the extension bit in the input and output of a stage respectively as shown in the example of drawing 11. As shown in drawing 13 and drawing 14, the clock of the input extension latch LEIN is carried out synchronous with the input data latch LDIN and the reasonableness test signal IN VALID. In order to refer to it easily, the various latches included in a doubleness stage constitute a pair with a respectively characteristic output signal.
[0232]In a doubleness stage, the output from the data latch LDIN forms the intermediate data called MID data. This middle data word is restricted when the middle acceptance signal (see with "MID ACCEPT" in drawing 13 and drawing 14) is set to HIGH, and it is loaded to the data output latch LDOUT.
[0233]The circuit part under the acceptance latches LAIN and LAOUT who show drawing 13 and drawing 14 is a circuit added to a fundamental pipeline structure, in order to generate various internal control signals used in order to double data. The NOT DUPLICATE signal used in order that the circuit concerned may control doubleness of the "data token" signal which shows that an effective data token is under processing, and data is included in these signals. When a circuit is processing a data token, a NOT DUPLICATE signal is toggled between a HIGH state and a LOW state, and makes each word in the token concerned double once (it restricts at a time) by this. When a data token with an effective circuit is under processing, a NOT DUPLICATE signal is held at a HIGH state. Therefore, this means that the token word

currently processed does not double.
[0234]As shown in drawing 13 and drawing 14, the input from output signal QI1 from top 6-bit and latch LI1 of 8-bit middle data word and a form input to the group of logic gate NOR1, NOR2, and NAND18 is formed. The output signal from gate NAND18 is displayed as S1. It can be shown using well-known Boolean algebra that output signal QI1 is "1", it restricts to a suitable distance with the structure which MID data word shows below, and the signal S1 is "0." All of "000001xx", i.e., top 5 bits, are "0", bit MID data [2] is "1", and the bit and MID data [0] position in MID data [1] have all any value. Therefore, the signal S1 has the structure where the MID data signal was planned, when the output from latch LI1 is "1", it restricts it, and it acts as a "token recognition signal" which is a low. The character of latch LI1 and its output QI1 is explained further below.
[0235]Latch LO1 carries out the function which latches the value of the last of a middle extension bit (it displays as "MID EXTN" and signal S4), and it loads this value to latch LI1 in that next rising edge of clock phase PH0. This latch's output is bit QI1 and is one of the inputs to the token decryption logical group who forms the signal S1. As already explained, the signal S1 merely descends to "0", when signal QI1 is "1" (and a MID data signal has the planned structure). Therefore, it is shown that the signal S1 merely descended to "0" always when the last extension bit was "0", and the front token ended it. Therefore, MID data word is the 1st data word in a new token.
[0236]The latches LO2 and L12 form the memory storage for a signal data token with NAND gates NAND20 and NAND22. On normal conditions, both of the signal S1 in the input to the signals QI1 and NAND22 in the input of NAND20 are logic "1." The art of Boolean algebra shows again that these NAND gates operate in the same method as an inverter on this condition. That is, in NAND20, it is reversed, and then this signal is again reversed by NAND22, and the signal Q12 from the output of latch LI2 forms the signal S2. In this case, since this passage has two logical reversal, the signal S2 has the same value as QI2.
[0237]It turns out like [how] that the signal data token in the output of latch LO2 forms the input of LI2. As a result, as long as it stops at the conditions that both QH1 and S1 are HIGH(s), a signal data token holds the state ("0" or "1"). This is truth also when the clock signals PH0 and PH1 are carrying out the clock of the latch (respectively L12 and LO2). When one or the both sides of the signals QI1 and S1 is "0", the value of a data token can only change.
[0238]Signal QI1 is "0" when a front extension bit is "0", as already explained at an early stage. Therefore, when a NID data value is the 1st word of a token, the signal of the first half is "0" always (and the address field for the tokens concerned is included). On this condition, the signal S1 is either "0" or "1." As already explained at an early stage, when MID data word has the planned structure which shows a data token in this example, the signal S1 is "0." When MID data word has other structures, (it is shown

that the tokens concerned are other tokens which are not data tokens), and S1 are "1."

[0239]When QI1 is "0" and S1 is "1", this shows that there are tokens other than a data token. The output of NAND20 is "1" as well-known in the field of digital electronics. NAND gate NAND22 reverses this (it already explained like), and the signal S2 is "0." As a result, this "0" value is loaded to latch LO2 when starting that following PH1 clock phase, and a data token signal is set to "0", and it is shown that the circuit concerned is not during processing of a data token.

[0240]QI1 is "0" and the signal S2 is "1" (when S0 is "0" and this shows a data token, and in spite of being other inputs of NAND22 from the output of NAND20). As a result, this "1" value is loaded to latch LO2 when starting that following PH1 clock phase, and it is shown that a data token signal is set to "1" and the circuit concerned is processing a data token.

[0241]A NOT DUPLICATE signal (output signal QO3) is loaded to the latch L13 in the next rising edge of clock PH0. Output signal QI3 from latch LI3 is combined with the output signal Q12 in gate NAND24 in order to form the signal S3. Like before, using Boolean algebra, it restricts, when it is a value "1" of signal QI2 and QI3 both, and it turns out that the signal S3 is "0." The signal 53 is set to "1" when signal QI2 is set to "0" (i.e., when a data token signal is "0"). If it puts in another way, when there will be no effective data token (QI2=0), or when data word is not doubleness (QI3=0), the signal S3 becomes a high.

[0242]Here, a data token signal assumes that it is still HIGH covering one or more clock signals. A NOT DUPLICATE signal (QO3) "is fed back" back to latch LI3, is that which is reversed by gate NAND24 (the input QI2 [another] is held at HIGH), and toggles output signal QO3 between "0" and "1." However, when there is no effective data token, signal QI2 is "0", and HIGH is forced into the signal S3 and output QO3 until a DATE token signal is set to "1" once again. output QO3 (NOT DUPLICATE signal) being fed back similarly, and, It restricts, when the value of both signals QA1 and QO3 is "1", and it is combined with output QA1 from the acceptance latch LAIN in a series of logic gates [as / those outputs of whose are "1"] (NAND16 and INV16 which form an AND gate together).As shown in drawing 13 and drawing 14, the output from an AND gate (gate NAND16 which resembles gate INV16 and therefore follows) is accepted similarly, and forms the signal IN ACCEPT. This acceptance signal is used in 2 line interface structure, as already stated.

[0243]The acceptance signal IN ACCEPT is used as a-izing signal to the latches LDIN, LEIN, and LVIN which can be operated. As a result, when a NOT DUPLICATE signal is a low, the acceptance signal IN ACCEPT is a low and the value which these three latches of all were made incompetent, and was memorized by those outputs is held. A stage does not receive new data until a NOT DUPLICATE signal is set to HIGH. This is added to the above-mentioned necessary condition, in order to force a high into the

output from the acceptance latch LAIN.
[0244]there is an effective data token (data token signal QO2 is "1") -- it restricts and signal QO3 is toggled between a HIGH state and a LOW state. As a result, an input latch is operation-ized, and it continues that it is long during the perfect cycle in every other [of both clock phase PH0 and PH1] one, and data acceptance is attained. Of course, the additional conditions referred to as that data acceptance preparation of the following stage is completed must still be satisfied so that it may be directed by a "HIGH" OUT ACCEPT signal. Therefore, it continues during a total of at least two clock cycles, and output latch LDOUT arranges the same data word to output bus OUT data. An effective data token (QO2 HIGH) carries out the whereabouts, when the reasonableness test signal QVOUT is HIGH, it restricts, and an OUT VALID signal is "1."
[0245]The signal QEIN which is an extension bit corresponding to MID data is combined with the signal S3 in a series of logic gates (INV10 and NAND10), in order to form signal S4. Each data word MID data is repeated by loading it to output latch LDOUT twice during the display period of a data token. "1" is forced into S4 by operation of NAND10 during these 1st period. Signal S4 is loaded to the latch LEOUT, in order to form OUTEXTN, at the same time as MID data is loaded to LDOUT, in order to form OUT data [7:0].
[0246]Thus, although predetermined MID data is loaded to LEOUT by the beginning and a high is forced into related OUTEXTN, to the 2nd, OUTEXTN is the same as the signal QEIN. Here, that QEIN is a low decides to consider the conditions in the period of the last word of one token which is known. MID data is loaded to LDOUT during the first period, and OUTEXTN is "1", and during the 2nd period, OUTEXTN is "0" and shows the end of the truth of the token concerned.
[0247]In the same gate combination (INVI2 and NANDI2), the output signal QVIN from the reasonableness test latch LVIN is combined with signal QI3, in order to form the signal S5. when the reasonableness test signal QVIN is HIGH by using well-known Boolean art, it turns out that the signal S5 is HIGH at one in case signal QI3 is a low (it is shown that data is doubleness) of cases. While MID data is loaded to LDOUT and a middle extension bit (signal S4) is loaded to LEOUT, the signal S5 is loaded to reasonableness test output latch LVOUT. Similarly, the signal S5 is combined with signal QO2 (data token signal) in the logic gates NAND30 and INV30 in order to form the output reasonableness test signal OUT VALID. As already stated at an early stage, there is an effective token, and when the reasonableness test signal QVOUT is a high, it restricts, and OUT VALID is HIGH.
[0248]In this invention, a MID ACCEPT signal, It is combined with the signal S5 in a series of logic gates (NAND26 and INV26) which carry out a well-known AND function in order to form the signal S6 used to the latch LOI, LO2, and LO3 as one of two-izing signals which can be operated. A MID ACCEPT signal is HIGH, the reasonableness

test signal QVIN is a high, or a token is doubleness, or (QI3 is ″0″) it obtains and shifts and the signal S6 goes up to ″1″ in that case. When the signal MID ACCEPT is HIGH and effective input data is loaded in the input of a stage, when clock signal PH1 is a high, or when the latched data is doubleness, latch LO1-LO3 is made operation possible always.
[0249]The data between stages is received from the above examination under control of the stage shown in drawing 13 and drawing 14 of a reasonableness test and an acceptance signal, and it turns out that it transmits. However, it is an exception to be combined with the doubleness signal which the output signal from the acceptance latch LAIN in an input side toggles like [in the case of a front example], as a result data word is twice outputted, before accepting a new word.
[0250]For example, of course, various logic gates like NAND16 and INV16 can be replaced by an equivalent logic circuit (in this case, one single AND gate). Similarly, when the latches' LEIN and LVIN output is being reversed for example, the inverters INV10 and INV12 are not required. Instead, the corresponding input to the gates NAND10 and NAND12 can be coupled directly with the output which has reversed these latches. As long as a suitable logical operation is carried out, the stage operates in the same method. Data word and an extension bit are still doubled.
[0251]the 1st data word of a token arranges ″1″ in the 3rd position of the word concerned, and it is cautious of the duplication function which the stage shown in a figure carries out not being carried out until it arranges ″0″ to five high bits, and is -- ** (Change and setting out are easily possible for the pattern needed by, of course selecting the connection of those other than NOR1 and NOR2 which are shown in other logic gates and figures, and NND18 gate.)
As shown in drawing 13 and drawing 14, unless the 1st data word has the structure of description, an OUT VALID signal continues during the whole token, and a low is forced into it. This demonstrates an effect which all the tokens except one token which makes a doubleness process cause make delete from the flow of a token. This is because the device connected to the output terminal (OUT data, OUTEXTN, and OUT VALID) does not recognize these tokens as effective data. The reasonableness test latches LVIN and LVOUT of the both sides in the stage concerned like before. It is possible to be reset using the reset signal back spread by one single reset input R of the one single conductor NOTRESETO and the downstream latch LVOUT so that a low may be made to force into an upstream reasonableness test latch in the following clock cycle.
[0252]In the example shown in drawing 13 and drawing 14, doubleness of the data contained in a data token, Only note serving only as a example of more data separates from a pipeline stage rather than are operational and a circuit reaches the input concerned as the result in ACCEPT and a VALID signal. In the example shown in drawing 13 and drawing 14, all the non-data tokens are removed for only explaining

how to show that similarly it is [circuit] operational in a VALID signal in order to remove data from a flow. However, in the most typical application, a pipeline stage is supplied simply, without correcting all the tokens that the pipeline stage concerned does not recognize. As a result, a pipeline's stage of everything but the lower stream further is able to act on the aforementioned token if needed. Drawing 15 and drawing 16 show an example of the timing diagram for the data doubleness circuit shown in drawing 13 and drawing 14. This timing diagram carries out the clock of the data between 2 phase clock signal, various insides, an external control signal and the input side of a stage, and an output side, and shows the relation between the methods of doubling data. now, it supposes that drawing 17 is referred to still in detail, and the process stage which is been alike and based on one mode of present this invention and which can be reconstructed is shown in this figure here.
[0253]The input latch 34 receives an input via the 1st bus 31. The 1st output from the input latch 34 is supplied to the token decoding subsystem 33 via the line 32. The 2nd output from the input latch 34 is supplied to the processing unit 36 as the 1st input via the line 35. The 1st output from the token decoding 33 is supplied to the processing unit 36 as the 2nd input via the line 37. The 2nd output from the token decoding 33 is supplied to the action identification unit 39 via the line 40. The action identification unit 39 receives inputs from the registers 43 and 44 via the line 46. The registers 43 and 44 hold the state of a machine as a whole. This state is determined by the history of the token received before. The output from the action identification unit 39 is supplied to the processing unit 36 as the 3rd input via the line 38. The output from the processing unit 36 is supplied to the output latch 41. The output from the output latch 41 is supplied via the 2nd bus 42.
[0254]Here, it supposes that drawing 18 is referred to and the start code detector (SCD) 51 receives an input via 2 line interface 52. This input is whether it is in the form of a data token, or to be by the data bit in data flow. The 1st output from the start code detector 51 is supplied to the 1st logic FIFO buffer (FIFO) 54 via the line 53. The output from the 1FIFO54 is logically supplied to the Huffman decoder 56 as the 1st input via the line 55. The 2nd output from the start code detector 51 is supplied to the DRAM interface 58 as the 1st input via the line 57. The DRAM interface 58 receives an input from the buffer manager 59 via the line 60. A signal passes the line 61, and is transmitted and received by the DRAM interface 58 to external DRAM (not shown). The 1st output from the DRAM interface 58 is supplied to the Huffman decoder 56 as the 1st physics input via the line 62. The output from the Huffman decoder 56 is turned to an index via line 63 ** as an input even to the data (ITOD) 64. The Huffman decoder 56 and ITOD64 operate together as one Logical unit. The output from ITOD64 is supplied to the arithmetic and logic unit (ALU) 66 via the line 65. The ALU66 to 1st output is supplied to the read-only memory (ROM) state machine 68 via the line 70. The output from the ROM state machine 68 is supplied to the Huffman

decoder 56 as the 2nd physics input via the line 69. The ALU66 to 2nd output is supplied to the token format part (TF) 71 via the line 70.

[0255]The 1st output 71 from TF concerning this invention is supplied to the 2nd FIFO73 via the line 72. The output from the 2nd FIFO73 is supplied to the reverse modeler 75 via the line 74 as the 1st input. The T/F71 to 2nd output is supplied to the DRAM interface 58 as the 3rd input via the line 76. The 3rd output from the DRAM interface 58 is supplied to the reverse modeler 75 as the 2nd input via the line 77. The output from the reverse modeler 75 is supplied via the line 78 as an input to the inverse quantization device 79. The output from the inverse quantization device 79 is supplied to reverse ZIG ZAG 81 (IZZ) as a reverse input via the line 80. The output from IZZ81 is supplied as an input to the discrete inverse cosine transform 83 (IDCT) via the line 82. The output from IDCT83 is supplied to a time decoder (drawing 19) via the line 84.

[0256]Here, it supposes that drawing 19 is referred to still in detail, and the time decoder based on this invention is shown in this figure. The fork 91 receives the output from IDCT83 (drawing 18) as an input via the line 92. As the 1st output from the fork 91, control tokens, such as a motion vector, are supplied to the address generator 94 via the line 93, for example. A data token is the purpose to count and is supplied to the address generator 94. This data is supplied to FIFO96 via the line 95 as the 2nd output from the fork 91. Next, the output from FIFO96 is supplied to the adding machine 98 as the 1st input via the line 97. The output from the address generator 94 is supplied to the DRAM interface 100 via the line 99 as the 1st input. A signal passes the line 91, and is transmitted and received by the DRAM interface 100 to external DRAM (not shown). The 1st output from the DRAM interface 100 is supplied to the prediction filter 103 via the line 102. The output from the prediction filter 103 is supplied to the adding machine 98 as the 2nd input via the line 104. The 1st output from the adding machine 98 is supplied to the output selector 106 via the line 105. The 2nd output from the adding machine 98 is supplied to the DRAM interface 100 as the 2nd input via the line 107. The 2nd output from the DRAM interface 100 is supplied to the output selector 104 via the line 108 as the 2nd input. The output from the output selector 104 is supplied to an animation formatting part (drawing 20) via the line 109.

[0257]In here, it supposes that drawing 20 is referred to and the fork 111 receives the input from the output selector 106 (drawing 19) via the line 112. As the 1st output from the fork 111, a control token is supplied to the address generator 114 via the line 113. The output from the address generator 114 is supplied to the DRAM interface 116 via the line 115 as the 1st input. The data as the 2nd output from the fork 111 is supplied to the DRAM interface 116 as the 2nd input via the line 117. A signal passes the line 118, and is transmitted and received by the DRAM interface 116 to external DRAM (not shown). The output from the DRAM interface 116 is supplied to the display

pipe 120 via the line 119.
[0258]As for each line, it is clear from the above-mentioned explanation not to interfere if needed, even if it has two or more lines.
[0259]In here, it supposes that drawing 21 is referred to and the one picture 131 is coded as one or more slices 132 in an MPEG standard. Each slice 132 has two or more blocks 133, and is coded by the right from the left for every sequence in each sequence. As shown in a figure, even if Sepang of each slice 132 is 132 one line or is the multi-line C of B less than one line in the block 133, C, or the block 133, it does not interfere correctly in the block 133.
[0260]It supposes that drawing 22 is referred to, a common intermediate format (CIF) is used in JPEG and an H.261 standard, and the picture 141 is coded as six rows in which each sequence includes two groups' block (GOB) 142. GOB142 is constituted by three rows or a number of blocks 143 with which six rows of either are not limited. Each GOB142 is zigzag coded in the direction of Z type shown by the arrow 144. As a result, GOB142 is processed toward the right in each sequence from the left for every sequence.
[0261]In here, it turns out that it supposes that drawing 23 is referred to and the output of an encoder takes the form of the data stream 151 about both MPEG and CIF. A decoder receives this data stream 151. Next, the decoder can restore an image according to the format used in order to code an image. In order for a decoder to enable recognition of the start and end point about each standard, 33 blocks of data streams 151 are divided into the length of 152.
[0262]The Venn diagram shown in drawing 24 expresses the field of a table selection possible value from the Huffman decoder 56 (drawing 18) of this invention. These values are duplication possible values to an MPEG decoder and an H.261 decoder. One single table choice shows specific MPEG and that both specific H.261 formats are decrypted.
Similarly, to an MPEG decoder and a JPEG decoder, these values are duplication possible values and show specific MPEG and that both specific JPEG formats are decrypted by one single table choice. An H.26l. value and a JPEG value do not overlap, but it is shown that single table selection which decrypts both formats independently does not exist.
[0263]Now, it supposes that drawing 25 is referred to still in detail, and this figure shows the outline of variable-length picture data based on operation of this invention. The 1st picture 161 that should be processed has the 1st picture start token 162, the 1st picture data 163 to which length is not limited, 1st picture -, and the token 164. The 2nd picture 165 that should be processed has the 2nd picture start token 166, the 2nd picture data 167 to which length is not limited, 2nd picture -, and the token 168. The picture start tokens 162 and 166 show the start to the processor of the pictures 161 and 165. Similarly, picture - and the tokens 164 and 168 mean the end to the

processor of the pictures 161 and 165. Thereby, a processor becomes possible [processing the picture images 163 and 167 of variable length].
[0264]In drawing 26, the split 171 receives an input via the line 172. The 1st output from the split 171 is supplied to the address generator 174 via the line 173. The address generated by the address generator 174 is supplied to the DRAM interface 176 via the line 175. A signal passes the line 177, and is transmitted and received by the DRAM interface 176 to external DRAM (not shown). The 1st output from the DRAM interface 176 is supplied to the prediction filter 179 via the line 178. The output from the prediction filter 179 is supplied to the adding machine 181 via the line 180 as the 1st input. The 2nd output from the split 171 is supplied as an input to the FIFO buffer (FIFO) 183 via the line 182. The output from FIFO183 is supplied as the 2nd input to the adding machine 181 via the line 184. The output from the adding machine 181 is supplied to the writing signal generator 186 via the line 185. The 1st output from the writing signal generator 186 is supplied to the DRAM interface 176 via the line 187. The 2nd output from the writing signal generator 186 is supplied as the 1st input to the read signal generator 189 via the line 188. The 2nd output from the DRAM interface 176 is supplied as the 2nd input to the read signal generator 189 via the line 190. The signal from the read signal generator 189 is supplied to animation formatting (not shown to drawing 26) via the line 191.
[0265]Drawing 27 shows a prediction filter process. The forward prediction filter 201 is supplied to the adding machine 203 via 202 as the 1st input. The backward prediction filter 204 is supplied to the adding machine 203 via the line 205 as the 2nd input. The output from the adding machine 203 is supplied via the line 206. As shown in drawing 28, the slice 211 has 1 or two or more macro blocks 212. As a result, each macro block 212 has four luminance luminosity blocks 213 and two chrominance blocks 214, and has the information about 16x16 blocks of the basis of a pixel. Each size of four luminance blocks 213 and two chrominance blocks 214 is 8x8 pixels. Four luminance blocks 213 are 1-pixel pixels. It has mapping in every pixel of the 16x16-block luminance (Y) information on the basis of a pixel. One chrominance block 214 includes the display of the chrominance level of a blue signal (Cu/b), and one chrominance block 21 includes the display of the chrominance level of a red signal (Cv/r) now. Subsampling of each chrominance level is carried out so that each 8x8 chrominance block 214 may contain the chrominance level of the 16x16-block chrominance signal of the whole of the basis of a pixel.
[0266]By referring to drawing 29, the structure and the function of the start code detector should become clear. The value register 221 receives image data via the line 222. The width of the line 222 is 8 bits and makes an 8-bit parallel transmission possible at once. The output from the value register 222 is supplied to the decoding register 224 in series via the line 223. The 1st output from the decoding register 224 is supplied to the detector 225 via the line 226. The width of the line 226 is 24 bits and

makes a 24-bit parallel transmission possible at once. The detector 225 detects the existence of an image or lack corresponding to the start code which became independent of the standard of "zero" value of 23 which follows by one "1" value. 8 bit-data value image follows an effective start code image. If existence of a start code image is detected, the detector 225 will transmit a start image to the value decoder 228 via the line 227.

[0267]The 2nd output from the decoding register 224 is supplied to the value decoding shift register 230 in series via the line 229. The value decoding shift register 230 can hold one data value image of the bit length 15. 8 bit-data value which follows a start code image is shifted to the right of the value decoding shift register 230 so that it may be shown by the subregion 231. This process eliminates duplication of a start code image so that it may inquire below. The 1st output from the value decoding shift register 230 is supplied to the value decoder 228 via the line 232. The width of the line 232 is 15 bits and makes a 15-bit parallel transmission possible at once. The value decoder 228 decrypts a value image using the 1st look-up table (not shown). The 2nd output from the value decoding shift register 230 is supplied to the value decoder 228 which supplies a flag to an index / token converter 234 via the line 235. The value decoder 228 supplies information to an index / token converter 234 via the line 236. Information is either a data value image which comes to hand from the 1st look-up table, or a start code index image. A flag shows the form in which information is supplied. The width of the line 236 is 15 bits and makes a 15-bit parallel transmission possible at once. Although 15 bits was selected as width which can be set in this case in this invention, not interfering, even if it uses the bit of other length should be understood. An index / token converter 234 changes information into a token image using the 2nd same look-up table as the case of a user's manual given in Table 12-3 (not shown). The token image generated by the yne deck / token SUKON barter 234 is outputted via the line 237. The width of the line 237 is 15 bits and a 15-bit parallel transmission is possible for it at once.

[0268]In drawing 30, the data stream 241 which comprises each bit 242 is inputted into a start code detector (drawing 30 is not shown). The 1st start code image 243 is detected by a start code detector. Next, a start code detector receives the 1st data value image 244. Before processing the 1st data value image 244, a start code detector can detect the 2nd start code image 245 which overlaps with the 1st data value image 244 in the length 246. If this occurs, a start code detector will not process the 1st data value image 244, and will instead receive and process the 2nd data value image 247.

[0269]In drawing 31, the flag generator 251 receives the data as the 1st input via the line 252. the width of the line 252 is 15 bits and is possible in a 15-bit parallel transmission at once -- it is alike and carries out. Similarly, the flag generator 251 receives a flag as the 2nd input via the line 253, and receives an input-signal-existing

effect image via 1st 2 line interface 254. The 1st output from the flag generator 251 is supplied to an input-signal-existing effect register (not shown) via the line 255. The 2nd output from the flag generator 521 is supplied to the decoding index 257 via the line 256. The decoding index 257 generates the following four outputs. That is, a picture start image is outputted via the line 258, a picture number image is outputted via the line 259, and an insertion image is outputted via the line 260, and a replacement image is outputted via the line 261. The data from the flag generator 251 is supplied via the line 262a. The header generator 263 uses a look-up table for ** which generates a replacement image, and this replacement image is supplied via 262b. In order that the extra word generator 264 may generate an insertion image, MPU is used, and an insertion image is supplied via the line 262c. The line 262a is combined in order to form 262 lines which is the 1st input to the output latch 265. The output latch 265 outputs data via the line 266. The width of the line 266 is 15 bits and makes a 15-bit parallel transmission possible at once.
[0270]An input-signal-existing effect register (not shown) supplies an image as the 1st input to 1st OR gate 267 via the line 268. An insertion image is supplied as the 2nd input to 1st OR gate 267 via the line 269. The output from 1st OR gate 267 is supplied as the 1st input to 1st AND gate 270 via the line 271. The logical NOT of a removal image is supplied as the 2nd input to 1st AND gate 270 via the line 272, and is supplied as the 2nd input to the output latch 265 via the line 273. The output latch 265 supplies an output effective image via 2nd 2 line interface 274. An output acceptance image is received by the output acceptance latch 275 via 2nd 2 line interface 274. The output from the output acceptance latch 275 is supplied to an output acceptance register (not shown) via the line 276.
[0271]An output acceptance register (not shown) supplies an image to 2nd OR gate 277 as the 1st input via the line 278. The logical NOT of the output from an input-signal-existing effect register is supplied as the 2nd input to 2nd OR gate 277 via the line 279. A removal image is supplied to 2nd OR gate 277 as the 3rd input via the line 280. The output from 2nd OR gate 277 is supplied as the 1st input to 2nd AND gate 281 via the line 282. The logical NOT of an insertion image is supplied as the 2nd input to 2nd AND gate 281 via the line 283. The output from 2nd AND gate 281 is supplied to the input acceptance latch 285 via the line 284. The output from the input acceptance latch 285 is supplied via 1st 2 line interface 254.
Table 600 format Reception picture Token . 1 H.261 sequence start Sequence start . MPEG picture start group start . Those without JPEG Picture start Picture data With no 2 H.261 Picture end Those without MPEG Padding Those without JPEG Flash plate The relation between the existence of a standard signal and lack in a stop after picture specific machine-independent control token. As shown in shown Table 600, detection of the image by the start code detector 51 generates a series of machine-independent control tokens. Each image listed by the "reception image"

column starts generation of all the machine-independent control tokens listed by the group of the "generation token" column. or [that a "sequence start" image is received during an H.261 processing term as shown in the 1st line of Table 600] -- or, When a "picture start" image is received during a PEC processing term, all the groups of four control tokens are generated, and each follows by one or more of the corresponding data value always. As shown in the 2nd line of Table 600, the 2nd group of four control tokens is generated at suitable time irrespective of the image received by the start code detector 51.
Table 601 display order I1 B-2 B3 P4 B5 B6 P7 B8 B9 I10 transmission order I1, as shown in one line of the table 601 showing the timing relationship between the pictures displayed as the picture transmitted P4 B-2 B3 P7 B5 B6 I10B8 B9, A picture frame is displayed by the numerical order. However, in order to decrease the number of the frames which must be memorized in a memory, a frame is transmitted in a different order. It is useful to begin analysis from intra-frame (I frame) one. I1 frame is transmitted to an order which should be displayed. Next, the frame (p frames) P4 the next is predicted to be is transmitted. Next, all the frames (B frame) that should be displayed between I1 frame and P4 frame and that are interpolated in both directions are transmitted. Frame B-2 and B3 express these frames. Making a front frame (forward prediction) or future frame (backward prediction) by this refer to it on the transmitted B frame makes it possible. After transmitting all the B frames which should be displayed between I1 frame and P4 frame, P7 which is the following p frames is transmitted. Next, all the B frames corresponding to B5 and B6 which should be displayed between P4 and P7 frame are transmitted. Next, 110 which is the following I frame is transmitted. Eventually, all the B frames corresponding to the frames B8 and B9 which should be displayed between P7 and 110 frames are transmitted. Since all things [merely holding two frames in a memory at a stretch, in order to transmit a frame in this order] display the B frame of middle required, it does not need making transmission of the following p frames or the I frame stand by to a decoder.
[0272]If the information beyond this about the structure of this invention, operation and the feature, the purpose, and an advantage is the usual expert in the technical field concerned, it should become clear easily from a detailed description additionally continued about the evincive example of this invention. About these detailed contents, explanation is given to describe collectively into the section shown below, in order clear and to make it convenience.
[0273]1. processing stage 7. multiplex standard coding 8. multiplex standard processing circuit in which characteristic 6. reconstruction of the extension 4.RAM memory map 5. bit stream of the DE coding 3. animation of a multiplex standard configuration 2.JPEG still picture is possible . - the 2nd -- an operation mode 9. start code detector 10. token 11.DRAM interface 12. prediction filter 13. accessing register 14. microprocessor interface (MPI)

15. an MPI lead timing 16.MPI light timing 17. keyhole address location 18. picture and a 19. flashing operation 20. flash function 21. picture -- back stop 22. multiplex standard search mode 23. reverse modeler 24. inverse quantization device 25. . The Huffman decoder and the parser (parser) (parser)
26. As shown in the U.S. Pat. No. 5,212,742 specification of the cosine transformer 27. buffer manager 1. multiplex standard configuration above-mentioned with separate versatility, Since various compression standards, i.e., JPEG, MPEG, and H.261 are common knowledge, it does not repeat about the detailed specification of these standards here.
[0274]As reference was made before, various this inventions can elongate various picture data bit streams coded by differing. the output of the space decoder which operates independently in each different standard about coding -- or, the data supplied to the in-series output of the space decoder which is put together and operates, and the time decoder, [treat and (it is explained still in detail continuously -- as)] In order RIFOMAIINGU [to use it including other display systems containing the display or animation display system in a computer and / this output], the output formatting part of a certain form by which grade regulation was carried out is required. The method of realizing this thing [formatting] changes substantially between a coding standard and/or the type of the selected display.
[0275]The block of data by which formatting was carried out in the 1st example based on this invention as drawing 17 - drawing 19 were described above, i.e., the 1st decoder (space decoder), Or an address generator is used, in order to memorize the output from one of the combination of the 1st decoder (space decoder) and the 2nd decoder (time decoder), to write in the information by which the double sign was carried out to a memory and/or to write it out in order of a raster. The animation formatting part explained below provides the output signal combination covering the wide range.
[0276]In the desirable multiplex standard animation decoder example concerning this invention, a space decoder and a time decoder need to realize both an MPEG coding signal and an H.261 dynamic image code decoding system. When applying in a low coding data rate, using a small picture format, the DRAM interface of the device of these both sides can be constituted in order to lessen quantity of DRAM needed. Suppose that it relates to a DRAM interface and explains further below about reconstruction of these DRAMs. Generally, one single 4-megabyte DRAM is needed by each time decoder and the space decoder circuit.
[0277]The space decoder of this present carries out all the processings needed within one single picture. Thereby, the relative redundancy in one picture decreases.
[0278]A time decoder decreases the redundancy between the pictures which arrive after arrival of the picture concerned which serves as a subject related to the picture which arrives before arrival of the picture made into a subject, and the picture made

into a subject. One mode of a time decoder is providing the address decode network which uses the circuit of the smallest possible number and treats a complicated address function required in order to read the data relevant to all these pictures by a high speed and the improved accuracy.
[0279]As drawing 18 was explained above, data arrives via the start code detector preceded with the Huffman decoder and a parser, FIFO, the 2nd FIFO register, a reverse modeler, an inverse quantization device, reverse ZIG ZAG, and reverse DCT. It is not necessary to carry out the whereabouts of the two FIFO on a chip. In one example, data does not flow through FIFO which carries out the whereabouts on a chip. Data is supplied to a DRAM interface and, in both cases, the whereabouts of a FIFO-IN storage register and the FIFO-DUT register is carried out out of a chip. These registers that require operation independently thoroughly [a standard] are explained still in detail continuously.
[0280]The large majority of a subsystem and a stage who shows drawing 18 has actually been independent of the specific standard used.
The buffer manager 59 who generates an address for the DRAM interface 58 and a DRAM interface, the reverse modeler 75, 8 l. of reverse ZIG ZAG, and reverse DCT83 are included.
The unit which became independent of the standard in the Huffman decoder and a parser contains ALU66 and the token formatting part 71.
[0281]In drawing 19, the DRAM interface 100, the fork 91, the FIFO register 96, the adding machine 98, and the output selector 106 are contained in the unit which became independent of a standard. A standard subordinate unit is the prediction filter 103 which can be reconstructed, as both the different address generator 94 in H.261 and MPEC, H.261, and MPEG are suited. JPEG data flows through the whole machinery no changed.
[0282]Drawing 20 shows the high-level lineblock diagram of an animation formatting part chip. Almost all the portions of this chip are independently from a standard. Few items influenced by a standard are methods as which data is entered in DRAM in the case of different H.261 from MPEG or JPEG.
And in H.261, it is necessary to code no single pictures.
There is some timing information referred to as time base which provides some information about whether it means to display a picture when, and is treated also by the address generation type of the logic in an animation formatting part.
[0283]The remaining portion of the circuit embodied by the animation formatting part is independently thoroughly from the specific compression standard used including all the color space conversions, rise sampling filters, and all the gamma correction RAM.
[0284]In this compression standard, the start code detector of this invention must recognize the start code pattern in which versatility [in / to each of the standard concerned / in said detector / a bit stream] differs depending on a compression

standard. For example, H.261 has a 16-bit start code and MPEC uses 24-bit start code **** and the marker code in which JPEG differs from other start codes considerably. If a start code detector recognizes these different start codes, the operation will once become independent of a compression standard intrinsically. For example, many of operations are dramatically similar among three different compression standards as the circuit which recognizes the marker of various different categories being another.
[0285]The unit shown below is the state machine 518 (drawing 18) located in the Huffman decoder and a parser. In this case, actual circuitry is almost the same about each of three compression standards. The only element influenced by a standard in operation is actually a reset address of a machine. When a parser is only reset, a parser is jumped to a different address about each standard. There are actually four standards recognized. These standards are H.261, JPEG, MPEG, and other one standards.
In the case of the last standard, a parser goes into one code used for a test.
The programs by the microcode this fact has a the same circuit in almost all modes, and concerning each standard only differ. Therefore, one program is operating in H.261, and when one another different program is running, there is no duplication among these programs. This is truth also about JPEG.
It is the program which the 3rd became independent of thoroughly.

[0286]The unit shown below is the Huffman decoder 56 which functions with the data index unit 64. These two units carry out a common operation, in order to carry out Huffman decryption. In this case, the algorithm used for the Huffman decryption is the same regardless of a compression standard. Change is saying which table is used and saying whether the data which carries out ingress to the Huffman decoder being reversed. The state machine which understands some modes of the Huffman decoder itself and encoding specification is included. Such different operations answer the command which carries out ingress, and are selected from a parser state machine. The parser state machine operates by a different program to each of three compression standards, and when [at which the standard under operation is suited] it differs, it supplies a right command to the Huffman decoder.
[0287]The unit of the last on the chip depending on a compression standard is the inverse quantization device 79.
In this case, the mathematics which an inverse quantization device carries out differs to each of a different standard.
In this case, a coding standard token is decrypted and it is memorized under which standard the inverse quantization device 79 is under operation. Next, it occurs after the event concerned, and it occurs and brews before other coding standards, and all the succession data tokens without ** are treated by the method shown by the

coding standard memorized in the inverse quantization device. The table which makes reference in a detailed description shows a different parameter in a different standard, and shows which circuit corresponds to such different parameters or mathematics.
[0288]The address generation based on H.261 is not the same to each subsystem shown in drawing 19 and drawing 20. The address-generation method which generates an address to two FIFO, in front of the Huffman decoder shown in drawing 18 or the back, does not change according to a coding standard. Even if it is a case of H.261, the address generation which occurs on the chip concerned does not change. Intrinsically, the difference between these standards is a difference of MPEG and JPEG.
The organization of a macro block is stationed in the linear line which crosses a picture and is extended horizontally.
As shown in drawing 21, the 1st macro block A covers one line. The macro block B covers less than one line. The macro block C covers a multi-line. In MPEG, it is divided into a slice, and when one slice may be the one level line A, one slice may be a part of level line B or one slice is extended in the following line from one line, there may also be C. Each of these slices comprises a sequence of a macro block.
[0289]In the case of H.261, since a picture is divided into the group (GOB) of a block, organizations differ a little. The group of a block is equivalent to a three-row-high macro block, and width is equivalent to 11 macro blocks. In the case of a CIP picture, there is this kind of block group of 12. However, it does not organize so that it may be called other one on one. That is not right and about 2 group arrangement of the 6-block-high group is carried out. That is, 6GOB and level 2GOB are arranged vertically.
[0290]In other standards of all the, when carrying out addressing, as already shown, the address of the macro block is carried out to turn. More particularly, even the start of the following line is [in / an address advances along with a line and / the termination of a line] ****. In the case of H.261, an order of a block is the same as the order described in the group of a block, but most it is ZIG ZAG when moving to the next block group.
[0291]This invention provides the circuit for treating the influence of the above. It is the method of changing the address generation in a space decoder and an animation formatting part into H.261. When information is written down in DRAM, this is attained always. Since it is entered by the knowledge about the above-mentioned address-generation sequence, the place in RAM arranged physically is strictly [as the case of the MPEG picture of the same size] the same. Therefore, since physical arrangement of the information in a memory is the same as the case where it is an MPEG sequence, all the address generation circuits for performing reading from DRAM do not need to understand that it is a case of an H.261 standard, when forming prediction, for example. Thus, in all the cases, only entry of data is influenced.
[0292]In a time decoder, abstraction is performed to H.261 to which it seems that a

circuit is something different from the actually generated matter. Namely, each group of a block is extended notionally, as a result instead of the rectangle which is 11x3 macro blocks, height is one macro block and, as for a macro block, length is extended to a 33 blocks (drawing 23) block group. By carrying out like this, in order that a block may carry out a group ****** count, the completely same technique as the count technique used in a time decoder is used also as an object for MPFG.
[0293]A circuit is designed so that agreement may exist between the H.261 group of a block, and an MPEG slice. It is preceded with each block group by the slice start code when H.261 data is processed after a start code detector. It is preceded with the next block group by the following slice start code. In order to count through this structure, when a count is performed in a time decoder, it is visible as if height is one macro block and length was a group of 33 macro blocks. Similarly, although a circuit is counted every ten intervals, this is enough as it. A count of the 11th macro block or the 22nd macro block will reset some counters. Each macro block is counted up and this is attained by the easy circuit provided with another counter reset to zero when 11 was reached. A microcode examines about it and carries out the operation. All the circuits in the time decoder of this invention are independently intrinsically from a compression standard about physical arrangement of a macro block.
[0294]There is a table where a large number differ about the adaptability of a multiplex standard, and a table suitable since a circuit is adapted at the suitable time in a suitable standard is selected. Each standard has two or more tables, and selects a circuit from sets of a table at the arbitrary predetermined time. In one arbitrary standard, a circuit selects one table at a stretch, and selects another, another table at a stretch. In a different standard, a circuit selects a set of a different table. As already pointed out on the occasion of the examination about drawing 24, intersection of a certain grade is among these tables. For example, one of the tables used for MPEG is similarly used in JPEG. These tables are not the sets separated thoroughly. Drawing 24 shows an H.261 set, an MPEG set, and a JPEG set. Note that very large duplication is between an H.261 set and an MPEG set. The table which these standards use is completely common. Since slight duplication is between MPEG and JPEG and there is no duplication between H.261 and JPEG, these standards have a set of a thoroughly different table.
[0295]As already pointed out, most system units are independently from a compression standard. one certain unit -- a standard -- when independent, this kind of unit does not need to memorize which coding standard is being processed. a standard -- all the subordinate units have memorized the compression standard, when a coding standard token flows through these units. When coding / decrypted information is distributed to the whole machine in the 1st encoding specification and a machine is changing a standard, generally the machine before microprocessor control is carried out selects functioning according to an H.261 compression standard. At a

different place of the plurality in the machine which a compression standard is changing, MPU in the machine of this kind of front generates a signal. When MPU differs, it changes, and a flash plate is carried out through the whole pipeline.
[0296]According to this invention, this change of a compression standard can be easily dealt with by making change of a coding standard token cause in the start code detector arranged as the 1st unit in a pipeline. A token expresses that a certain coding standard is being started, and the control information concerned flows through a machine toward the lower stream, and other registers of all the are constituted at the suitable time. MPU does not need to program each register.
[0297]A prediction token tells the method of forming prediction using the bit in a bit stream. According to which compression standard is applying, a circuit changes into a prediction mode token the information discovered in the standard concerned from a bit stream. It is easy for this processing to be performed by the Huffman decoder and the parser state machine, and to operate the bit based on specific conditions in this case. A start code detector generates this prediction mode token. Next, a token flows through a machine toward the lower stream to the circuit of a time [to be a device which is responsible for prediction formation] decoder. Since the bit within a certain standard does not change into three different standards, the circuit of a space decoder interprets the token concerned, without getting to know which standard the standard to which the token is applied in it is. A space decoder performs only what was answered and told to the token concerned. The design of other units in a machine is simplified by using them appropriately by owning these tokens. Although the program can become a little complicated, the profits which can use hard wire logic with a difficult design in this case to a multiplex standard are obtained.
[0298]2. As pointed out before decryption of a JPEG still picture, more particularly, this invention relates to extension of a coding moving image signal unrelated to a use compression standard about signal extension.
[0299]the signal (MPEG.) with which the 1st was coded in the pipeline processing system as for one mode of this invention Or in order to decrypt an H.261 coded dynamic image signal, it is providing the 1st decoder circuit (space decoder) for decrypting the signal (JPEG coding moving image signal) with which the 1st was coded combining the 2nd decoder circuit (time decoder). A time decoder is not needed for JPEG numerals decryption.
[0300]By this invention, one single pipeline decoder and an extension system are used about this point.
Therefore, extension of two or more signals coded by a different method is made easy. Numerals decryption and an extension pipeline processor are organized by the special configuration which makes possible treatment of a multiplex standard coded dynamic image signal peculiar using a single pipeline decoder and a processing system, and the art that is thoroughly compatible. A space decoder is combined with a time decoder,

and it is used in order that an animation formatting part may drive an animation display.
[0301]Other modes of this invention are using the combination of a space decoder and an animation formatting part, in order to use it only with a still picture. In one boundary of a single picture, the space decoder which became independent of a compression standard carries out all the data processing. This kind of decoder treats space extension of the internal picture data distributed in the address generation circuit which became independent of the standard for passing a pipeline and treating memory and search of information to related random access memory and memory. Still picture information is decrypted in the output of a space decoder, and this output is used as an input of the multiplex standard composition good activity drawing formatting part which supplies an output to a display terminal. In the 1st sequence of the same picture, each elongated picture in the output of a space decoder is the same bit length until a picture reaches the output of a space decoder. When it does not interfere even if it was thoroughly different image size, therefore it compares with the 1st length, even if the 2nd sequence of a picture differs in length, it does not interfere. The whole of this kind of a similar picture of the 2nd sequence has the same bit length until this seed picture will reach the output of a space decoder, if it explains again.
[0302]Other modes of this invention are selected and systematized so that it may operate considering the standard subordinate bit stream which carries out ingress as a pipeline processor which became independent of a standard and which can be reconstructed, It is systematizing internally to a series of control tokens and data tokens which were combined with two or more processing stages which have been arranged systematically, and which can be reconstructed.
[0303]In chip outside, the 1 without DRAM ** single space decoder can decrypt a base-line JPEG image quickly about JPEG decryption. A space decoder supports all the functions of a base-line JPEG coding standard. However, the image size which can be decrypted may be restricted by the size of the equipped output buffer. A space decoder circuit includes the random-access-memory circuit which has the address generation circuit which became independent of the machinery subordination for treating the memory to the memory of information, and a standard.
[0304]A time decoder is not needed in order to decrypt the animation by which JPEG coding was carried out, as already pointed out. Therefore, direct supply of the signal carried by the data token is carried out through a time decoder, without processing more, when the time decoder is constituted as an object for JPEG operations.
[0305]Other modes of this invention are providing one pair of memory circuits like a buffer memory circuit for being combined with the Huffman decoder / animation demultiplexer circuits (HD&VDM), and operating in a space decoder. The 1st buffer storage is arranged in front of HD & VDM, and the 2nd buffer storage is arranged after HD & VDA. KD & VDM decrypts the binary 1 and the bit stream constituted 0 appearance. The whereabouts of these figures is carried out into the bit stream by

which standard coding was carried out, and the bit stream concerned is changed into the number used in the lower stream. The advantage of 2 buffer system is being able to use it in order to realize a multiplex extension system. These two buffers are explained in detail by next combining the confirmed example of the Huffman decoder.
[0306]Another mode of a Shigeki Honda rank expansion circuit is combined with the start code detector circuit arranged upstream of the 1st forward buffer that operates combining the Huffman decoder. One advantage of this combination is that the flexibility which must be applied to a bit stream increases, when treating an input bit stream, especially padding. Arrangement of these confirmed components, a start code detector, a memory buffer, and the Huffman decoder will strengthen the treatment of the specific sequence in an input bit stream. DRAM outside a chip is used in order to decrypt the animation picture by which JPEG coding was carried out in real time. It depends for the size and speed of a buffer which are used with DRAM on the dynamic-image-code-ized data rate.
[0307]Encoding specification identifies all the information required in order to memorize to DRAM relevant to the space decoder using the circuit which became independent of a standard standard subordinate type.
[0308]3. In animation extension this invention, when an animation is being elongated through a decryption process, a time decoder is still more nearly required. A time decoder combines with the decrypted picture already which is having it meant to display the data decrypted in the space decoder after the picture image decrypted now. A time decoder receives the information for identifying this information replaced in time in an image coding data stream. a time decoder carrying out the address of the information arranged in time and spatially, retrieving information, and being decrypted now, and, It completes and information is combined by a method which decrypts the information arranged in one picture using the picture which terminates the consequential picture which was suitable for transmission in an animation formatting part in order to drive a display screen. Instead, a consequential picture is memorizable in order to use it after that for time decryption of the picture which follows.
[0309]Generally, in either an early stage and/or the second half, a time decoder carries out processing between pictures to the picture decrypted now. A time decoder reintroduces the information which is not coded in the display by which the picture concerned was coded. The information of the reason concerned is redundant and it is already based on an available thing in a decoder. All predetermined pictures are able to, include similar information which surrounds the information concerned in time in both approximately more particularly. If motion compensation is applied, this similarity can be enlarged further. A time decoder and an expansion circuit decrease the relative redundancy between the related pictures similarly.
[0310]In other modes of this invention, a time decoder is used in order to treat the standard subordinate print-out from a space decoder. This standard subordinate

information about one single image, In order that the elongated print-out which was processed by the space decoder may combine one picture of the information packet by which the picture information which has been arranged in time about the time position of the 1st picture, and which was decrypted spatially was decrypted spatially, In a method which is memorized by other DRAM registers with other random access memory which has the address generation circuit which was subordinate to other machines and became independent of a standard, it is distributed among some fields of DRAM.

[0311]In the multiplex standard circuit which can decrypt the signal by which MPEG coding was carried out, it may be required that a graphics format that the still larger logic DRAM buffer can suit MPEG and also large should be supported.

[0312]Picture information is under movement through the in-series pipeline in 8 pixel x8 picture element block. In one form of this invention, an address decoding circuit treats these picture element blocks along a block border (it memorizes and searches). An address decoding circuit treats the memory and search of 8x8 picture element block which are performed by crossing a boundary. Suppose that it explains still more nearly thoroughly below about this flexibility.

[0313]Similarly, in order to treat without signal-processing delay, even if the 2nd time decoder supplies the output of the 1st decoder circuit (space decoder) to a direct animation formatting part, it does not interfere.

[0314]A time decoder resets the block of image data in order to delay by a display circuit. An address decoding circuit makes it feasible this to reset in order so that it may explain later.

[0315]the picture under processing by which one important feature of a time decoder was acquired from selection of the picture as already stated -- an early stage -- or it is it being behind and totaling the picture information which arrived.

[0316]When describing a picture in this context, a picture means either of the contents shown below.

[0317]1. It is the data display by which the picture was coded.

[0318]2. That is [it is a result], it is a picture which is obtained as a result of having added the process stage performed by the decoder and which has been decrypted [final].

[0319]3. It is the decrypted picture already read from DRAM.

[0320]4. It is a result of spatial decryption. That is, it is the range of picture start token and its following picture - and the data between tokens.

[0321]After picture data information is processed by the time decoder, information is displayed by the picture memory location or is returned. Next, this information is held as further standard used when processing another different coded data picture. For a visual display **** attained when the good necessary picture by which scramble was carried out changes the rearrangement function of a time decoder is included by the

rearrangement of the picture by which MPEG coding was carried out.
[0322]4. All of a RAM memory map space decoder, a time decoder, and an animation formatting part use external DRAM. It is preferred that the same DRAM is used for all these three devices. Even if all these three devices are the cases where all three devices use a DRAM interface with an address generator further using DRAM, what each realizes in DRAM is not the same. That is, even if each chip, for example, a space decoder, and a time decoder are the cases where the same physical external DRAM is used, they have a different DRAM interface and address generation circuit.
[0323]In short, a space decoder realizes two FIFO in common DRAM. It supposes that drawing 18 is referred to again, and one FIFO54 is arranged in front of the Huffman decoder 56 and a parser, and another side is arranged after the Huffman decoder and a parser. FIFO is realized in a comparatively direct method. The portion with special DRAM is deducted as a physical memory for realizing FIFO in it to each FIFO.
[0324]Two pointers are used for the address generator connected with the space decoder DRAM interface 58, and it manages the track of a FIFO address. One pointer points out the 1st word memorized by FIFO, and another pointer points to the word of the last memorized by FIFO. Therefore, reading / writing operation to a predetermined word are enabled. When the termination of a physical memory is reached in reading or an operation process [in / it writes and / operation], "the wrap around (wraps around)" of the address generator is carried out to the start of a physical memory.
[0325]In short, even if which encoding specification (MPEG or H.261) is specified, the time decoder concerning this invention must be able to memorize two perfect pictures or frames. in order to explain simply, the physical memory of DRAM which is going to memorize two frames in it shall be divided into two half portions, and it is considered as the exclusive use over specific one out of two (using a suitable pointer) pictures half a minute each, respectively.
[0326]Use the picture of three different types, namely, MPEG is Intra (I), prediction (P), and bidirectional interpolation (B). As already stated, B picture is based on the prediction from two pictures. As for one picture, one side is obtained from the past now from the future. When decrypting P and B picture, in order [although I picture does not need decryption beyond it by a time decoder, however] to use it later, it must be memorized by one of two picture buffers. It is required for decryption of P picture to form prediction from decrypted P already or I picture. Decrypted P picture is memorized by one picture in order to use for decryption of P and B picture. B picture can require the prediction from both picture buffers. However, B picture is memorized by external DRAM.
[0327]When I and P picture are decrypted, note not being outputted from a time decoder. Instead, I and P picture are entered in one of the picture buffers, when the following I or P picture arrives for decryption, they are restricted, and they are read. If it puts in another way, a time decoder will trust the following P or I picture, in order to

carry out the flash plate of the front picture from two picture buffers, so that it may be further explained after this section about flashing. In short, the space decoder can supply fake I or P in the termination of an animation sequence, in order to carry out the flash plate of P or the I picture. As a result, when the following animation sequence starts, the flash plate of this fake picture is carried out. Loading of peak memory band width occurs when decrypting B picture. The worst state is a case where the B frame is formed from this kind that all the prediction is created by the accuracy which is half a pixel, and is supplied from both picture buffers of prediction.
[0328]As described above, a time decoder can be constituted so that re-ordering of an MPEG picture may be provided. The output of P and I picture is delayed until decryption by that next P in a data stream or the time decoder of I picture starts by this picture re-ordering.
[0329]A specific token is temporarily memorized by the chip as a picture will be entered in a picture buffer, if P or I picture is re-set in order. If a picture is read to a display, these tokens memorized will be searched. In the output of a time decoder, the data token of P decrypted newly or I picture is exchanged for older P or I picture.
[0330]On the other hand, H.261 manufactures prediction only from the just decrypted picture. One of the two picture buffers is filled in, and it becomes usable to the next picture decryption as each picture is decrypted. The only DRAM memory operation needed is writing 8x8 blocks.
And it is forming prediction by the motion vector of integer accuracy.

[0331]In short, an animation formatting part memorizes three frames or a picture. Three pictures need to be memorized in order to accommodate a function which a picture skips repeatedly.
[0332]5. It is useful to re-evaluate the bit stream characteristic of the DEA stream which decided to refer to especially the space decoder concerning characteristic this invention of a bit stream, and was coded. This is because these characteristics must be treated by the circuit of a space decoder and a time decoder. For example, under one or more compression standards, the rate of a compression ratio of the standard concerned is attained by changing the number of the bits used in order to code two or more pictures of one picture. The number of bits may change [the wide range]. In detail, it is identifiable as one unit length in the length of the bit stream used in order that this may code the standard picture of one picture, and even if other pictures are the length of many units, they do not interfere, and it is possible [pictures] for the 3rd picture to be also a portion of the unit concerned.
[0333]When neither of the current standards (MPEG1.2, JPEG, H.261) defines how to end a picture image but the following picture starts it, it is tolerated that the present picture finishes. A standard (especially H.261) makes it possible to generate an imperfect picture by an encoder.

[0334]Based on this invention, the method of showing the end of one picture is provided by using 1 picture end of a token. The stillness coding data which separates from a UTATO code detector comprises the picture from which length changes substantially, although it starts by a picture start token and ends by picture - and a token. Although other information transmitted (between the 1st and the 2nd picture) may be here, it is known that the 1st picture was completed.
[0335]To a predetermined sequence, the data stream in the output of a space decoder has the same length (number of bits), and still comprises the picture of a picture start and a picture end. Even if the length of the time between a picture start and a picture end changes, it does not interfere.
[0336]In the fixed picture rate which receives these uneven pictures in time and is determined by the type of the display currently driven, an animation formatting part displays these on a screen. For example, a display rate which is variously different like a PAL-NTSC television standard is used all over the world. In a peculiar method, this is selectively attained a ZURU ping or by repeating in Kuche. Although it operates with ordinary "frame rate converter", for example, 2-3 pulldown **, and a fixed input picture rate, the animation formatting part can treat a variable input picture rate.
[0337]6. Supposing that processing stage re-****17 which can be reconstructed is referred to, the processing stage (RPS) which can be reconstructed has the token decode circuit 32 used in order to receive the token which carries out ingress from 2 line interface 33 and the input latch 34. The output of the token decode circuit 33 is supplied to the processing circuit 35 via 2 line interface 36 and the action discrimination decision circuit 39. The processing circuit 35 is suitable for data processing under control of an action discrimination decision circuit. After processing is completed, the processing circuit 36 connects to the output of 2 line interface bus 40 the signal completed via the output latch 41.
[0338]The action discernment decode circuit 39 has an input which is supplied from the token decode circuit 33 via 2 line interface bus 40, or is supplied from the memory circuits 43 and 44 via 2 line interface bus 46, respectively. The token from the token decode circuit 33 is simultaneously supplied to the action discrimination decision circuit 39 and the processing circuit 36. In this next portion of this specification, an action identifying function and RPS are explained still in detail using a table and figures.
[0339]The functional block diagram in drawing 17 shows the stage which is not the circuit which became independent of drawing 18, drawing 19, and the standard indicated to drawing 20. Data flow flows into 2 line interface circuit 42 via the output latch 41 via the processing stage 35 via the token decoder 32. When a control token is recognized by RPS, it is decrypted in the token decode circuit 33, and suitable action is performed. When not recognized, 2 line interface circuit 42 is supplied without change via the output circuit 41. The present invention functions as a pipeline

processor with 2 line wire interface circuit for controlling a motion of a control token via a pipeline. It is ending with an indication still in detail about this feature of this invention to the European patent application 92306038.No. 8 for which it applied before.
[0340]In this invention, the token decode circuit 33 is used in order to identify any of a data token or a control token the tokens in Iriki are now through 2 line interface 42. When it has been recognized that the token is investigated by the token decoder circuit 33, it is outputted to the action discrimination decision circuit 39 by the suitable index signal or flag signal which shows action which should be carried out. The token decode circuit 33 supplies a suitable flag or index signal to a processing circuit, and, simultaneously with it, warns of existence of the token currently treated by the action discrimination decision circuit 39 to a processing circuit. It does not interfere, even if a control token is processed similarly.
[0341]In this invention, a token various type [usable] is explained still in detail in next. It is enough that it is just cautious of the address carried by the control token being used about this portion of this specification in order to be decrypted in the decoder 33 and to access the register contained in the action discrimination decision circuit 39. In being a control token the token currently investigated has been recognized to be, in order to cover the whole state machine and to distribute a control signal, the reconstruction state circuit is used for the action discrimination decision circuit 39. As reference was already made, this operation-izes the state machine of an action discernment decoder, and this decoder reconstructs the decoder itself. For example, said decoder can change encoding specification. Thus, the action discrimination decision circuit 39 decrypts necessary action for treating the specific standard supplied via the state machine shown in drawing 17.
[0342]Similarly, the processing circuit 36 currently controlled by the action discrimination decision circuit 39 makes the preparation which processes the information which is included at the data field of a data token in a suitable case, in order to cause this phenomenon. In many cases, a control token reaches first, the action discrimination decision circuit 39 is reconstructed, and the data token processed by the processing circuit 36 next follows just the back. In the processing unit 36, a control token comes out of the output latch circuit 41 via the output 2 line interface 42 which precedes processed data token just a front.
[0343]In this invention, the action discrimination decision circuit 39 is a state machine holding the history State. The registers 43 and 44 are decrypted from the token decoder 33, and hold the information memorized by these registers. If needed, even if on chip or this kind of registers are any of an off-chip, they do not interfere. The State register of these plurality includes the action information identified in the action discrimination decision circuit 39 connected to action discernment now. This action information is memorized from the token decrypted before, and the thing which is

selected and which carry out action influence is possible for it. The connection 40 is linearly connected from the token decoding 33 to the action identification block 39. It means that it is shown that this may be influenced by the token by which action is being processed by the token decode circuit 33 now.
[0344]The token decryption and data processing based on this invention are shown. Data processing is performed so that it may be constituted by the action discrimination decision circuit 39. action being influenced by the number of conditions, and being influenced using the information acquired from the decrypted token already on the whole, or, The information memorized from the decrypted token already in the registers 43 and 44 more particularly, the present token under processing, and the action identification unit 39 are influenced by the State and history information which were acquired by itself. Distinction between a control token and a data token is shown by it.
[0345]In all RPSs, some tokens are seen as a control token with the RPS unit concerned. In this case, these tokens should influence the operation of RPS a little at the next time. A set of everything but a token is looked at by RPS as a data token. This kind of data token includes the information processed by RPS in the design of a specific circuit, a decrypted token already, and a method that is determined by the state of the action identification unit 39.
[0346]Although specific RPS identifies the set of a certain specific token and the set of other tokens to specific RPS control as data, this is a view of the specific RPS concerned. Another PPS can have another different view to the same token. In one side, some tokens are seen as a data token with one RPS unit, and can be determined with another RPS unit in another side as what is actually a control token. For example, quantization expression news is data at least in the limitation about the Huffman decoder and a state machine. This is because the information concerned is formed by the token called the quantization table token (QUANT table) which arrives as data coded by the input, and is form-ized by a series of 8 bit word, and moves a processing pipeline downstream. In the limitation about the machine concerned, all the information must have been data, the information concerned is handling data, a certain kind of data is changed into another kind of data, and this is a processing capability carried out by the portion concerned of the machine concerned clearly. However, if the information concerned reaches an inverse quantization device, an inverse quantization device will memorize the information in the token concerned which are two or more registers. There are the 8 numbers of bits of 64 pieces, and since it is many registers, many registers can actually exist as a whole. Since the number which this information is regarded as control information, therefore carries out the multiplication of the information to each data word is influenced, the control information concerned influences the processing performed in that following data token. It considers that one stage is data about the token concerned, and there is an

example regarded as other stages being control about the token concerned.
[0347]In almost all cases, it considers that the token data based on this invention is data which passes a machine. One of the important modes is looking for the set of the token with each stage of the circuit which has a token decoder specific generally.
And the token which the stage concerned does not recognize passes a stage, not changing, and is moving a pipeline downstream.
As a result, there is an advantage referred to as that the downstream stage which follows the present stage looks at these tokens, and these tokens can be answered. This is the important feature. That is, communication between blocks is possible by using a token mechanism to the block which does not adjoin.
[0348]Other important features of this invention each stage of a circuit, It is having throughput within limits which can perform required operation for each standard, and that control is executable about the operation which must be performed at the predetermined time as a token. In order to provide this capability, there is one different process element between different stages. There are three separated thoroughly different programs in state machine ROM of a parser. Each program is equivalent to each standard treated. It is dependent on a coding standard token which program is executed. The capability for each of these three programs to treat both decryption and coding standard standard tokens to the inside if it puts in another way is also **. When it is understood which is the encoding specification as which each of these programs should be interpreted next, these programs are jumped to a literal to the start address for the program concerned in microcode ROM. This is the way a stage treats multiplex standard nature.
[0349]Two matters are influenced by a different standard. Since the shift register for detecting the length of a start marker code to the 1st is reconstructed, it is influenced which pattern of the bit in a bit stream is recognized as a start code or a marker code. One information which displays what the start concerned or a marker code means on the 2nd is whereabouts **** in a microcode. Coding of a bit should remember differing among three standards. Therefore, refer to the type of what is independently from the standard standard concerned, i.e., the token showing an ingress code, for a microcode in a table peculiar to the compressor standard. Since each of various standards provides the specific code which the standard concerned generates when the most, generally this token is independently from the standard concerned.
[0350]The inverse quantization device 79 has mathematical capability. A quantizer performs multiplication and addition and has the capability to perform all three compression standards constituted with a parameter. For example, the flag bit in ROM in a control device tells whether the constant K is applied to an inverse quantization device. Another flag tells whether other constants are applied to an inverse quantization device. An inverse quantization device memorizes a coding standard token in a register, when a token flows with a quantizer. When a data token passes

after it, an inverse quantization device memorizes what the standard concerned is, and in order to perform suitable operation, it searches for the parameter which needs to be supplied to a process element. For example, an inverse quantization device looks for whether K is set to 0 to a specific compression standard, or it is set to 1, and supplies the result to the processing circuit. In the same meaning, the Huffman decoder 56 has many tables in it, the some are the objects for JPEG, the some are the objects for H.261, and the some are the objects for MPEG. The large majority of these tables is actually useful for one or more of the compression standards of these. It is dependent on the syntax of the standard concerned which table is used. The Huffman decoder operates by receiving the command from the state machine which tells which table is used. Therefore, the Huffman decoder is inputted into it, is memorized and does not have in itself one state of expressing which coding being carried out and, directly. It is the combination of a parser state machine and the Huffman decoder which becomes that is not right and together and includes information in them.
[0351]An address generation is converted, about the space decoder of this invention, when shown in drawing 17, it is similar, and much information is decrypted, for example from a token like encoding specification. Similarly, encoding specification and additional information are recorded in a register, and when the information concerned passes the macro block in a system and counts one by one, they influence advance of an address generator state machine. The last stage operates by either two mode H.261 or MPEG, and is the prediction filter 407-9 (drawing 26) identified easily.
[0352]7. Multiplex standard coding and also the control token based on this invention in detail have two or more words in the token concerned. In this case, as an extension bit, one bit known is set and use of the addition word in the token for carrying additional information is specified. A certain thing of these additional control bits contains the index which shows the information used in order to create the index signal which became independent of 1 set of standards in the corresponding state machine. The remaining portion of a token is used in order to show and identify the processing control facility of the inside which serves as a standard to all the data streams which pass a pipeline processor.
[0353]In that of one form of this invention, token extension, it being used in order to carry the present encoding specification decrypted by the relative token decoding circuit continued and distributed to the whole machine, and, And since the action discrimination decision circuit 39 of a stage covering the whole machine is reconstructed always when operating under new encoding specification is appropriate, it is used. The token coding circuit can show whether one and the control token of the selected standard which was designed so that the circuit concerned might treat are related.
[0354]More particularly, 8 bits of ** value follows an MPEG start code and a JPEG

marker. 4-bit value follows that of an H.261 start code. In this context, the start code detector 51 shows that the following 8 bits contain the value connected with the start code by detecting either an MPEG start code or a JPEG marker. Next, said detectors are any of an MPEG start code or a JPEG marker independently.
and the signal which shows that it is not an H.261 start code is created -- thing **** can be carried out.
8-bit value is inputted into a decode circuit in this 1st case, and a part of decode circuit creates the signal which shows the index and flag which are used in the present circuit, in order to treat the token which passes through the circuit concerned. This is set after that, and also in order to insert the portion of the control token referred to in order to determine which standard is being treated, it is used. In this meaning, a control token contains the portion which shows that it is related to an MPEG standard and the portion which shows the type of operation which acts on company data. As already inquired, since the processing stage used for a ****** sake in the function needed by various standards made for that purpose is reconstructed, this information is used in a system.
[0355]For example, this code is connected with the 4-bit value which follows just after a start code about an H.261 start code. A start code detector supplies this value to a token generator state machine. This value is supplied to eight bit decoders which make the number of triplet starts. The number of starts is used in order to identify the picture start of the number of pictures so that it may be shown by the value concerned. A system contains the multiplex parallel processing pipeline who operates under the principle of already described 2 line interface. Each stage contains the machine which takes the form generally shown in drawing 17. The token decoding stage 33 is used in order to lead the token which a state machine is made to enter suitably to the action discrimination decision circuit 39 or the processing unit 36. A processing unit is ending with reconstruction at a form required in order to treat the present encoding specification by the control token before the next, and current standards are being entered by the processing stage now, and are carried by the following data token. It is possible for the continuous state machine in a processing pipeline to function under one encoding specification, i.e., H.261, based on this mode of this invention, and, on the other hand, a front stage can be operated, for example under an individual standard like MPEG. In order to carry both a control token and a data token, the same 2 line interface is used.
[0356]The control token for which it is required that the system of this invention should decrypt much encoding specification with the processing stage which can reconstruct the fixed number is used. Since it is important to, direct the time of a picture actually being completed more particularly, picture - and a control token are used. Therefore, to design a multiplex standard machine, it is required to make the additional control token which shows which standard decryption art is used in a

multiplex standard pipeline. This kind of control tokens are picture – and a token. This picture – and a token are forced so that the flash plate of the buffer may be carried out, and they are used in order to show that the present picture was completed in order to extrude the present picture to a display via a decoder.
[0357]8. interconnection of the compression standard subordinate circuit of the form of the start code detector boiled and described by the multiplex standard processing circuit-2nd operation mode ** is suitably carried out to the circuit which became independent of a compression standard via a suitable bus. A standard subordinate circuit is connected to a subordinate independent combinational circuit via the same bus and additional bus. The circuit which became independent of a standard supplies an additional input to a standard subordinate independent circuit, and a standard subordinate independent circuit makes back offer of the information simultaneously in the circuit which became independent of a standard. The information from a circuit that it became independent of a standard is supplied to an output via other suitable buses. the inside of the bit stream each coded by the multiplex standard in which Table 600 is applied to the input to the standard subordinate start code detector 51 -- a standard -- it is shown that a specific bit stream with a subordinate meaning is contained.
[0358]9. As stated to start code ********, the start code detector based on this invention can treat MPEG, JPEG, and H.261 BITTOTO ream, and can generate a series of significant ownership tokens from said stream for the remaining decoders. If shown as an example by which decryption of a multiplex standard is attained, an MPEG (1 and 2) picture start code, An H.261 picture start code and a JPEG start OBUSU can (SOS) marker are treated as equivalent by a start code detector, and these [all] generate an internal picture start token. In the same method, as for an MPEG sequence start code and JPSG 501 (start buoy mage) marker, both generate a machine sequence start token. However, an H.261 standard does not have an equivalent start code. Therefore, a start code detector answers the 1st H.261 picture start code, and generates a sequence start token.
[0359]No above-mentioned images are directly used in addition to SCD. Rather, it seems that a machine picture start token is equivalent to the picture start image included in a bit stream, for example. He would like to bear the following matter in mind, namely, the machine picture start itself is not a direct image of the picture start in a standard. Rather, it is a control token used combining other control tokens, in order to provide the decoding function which became independent of the standard which emulates the operation of the image in each compressed code-ized standard. The combination with the combination of the control token combined with the reconstruction of the circuit based on the information carried by the control token and index, and/or flag which were further generated by the token decoding circuit portion of each state machine is peculiar. The state machine in which typical

reconstruction is possible is explained following the next.
[0360]It supposes again that Table 600 is referred to, and the name of the group of a standard image is shown in a left column. The machine subordinate control token used for the emulation of the signal which does not exist in the standard image concerned or is not used, and by which standard coding was carried out is shown in the right column.
[0361]In Table 600, as already stated, when decrypting one with an arbitrary standard signal indicated in Table 600, it turns out that a machine sequence start signal is generated by a start code detector. A start code detector makes a sequence start, a group start, a sequence and a slice start, user data, extra data, and a picture start token, in order to supply 2 line interface continued and used for the whole system. Each of these stages that operate with these control tokens, It is constituted by the index which is constituted by the contents of the token or is made by the contents of the token, and preparation treating the data predicted to be received when a picture data token arrives at the station concerned is made.
[0362]As already stated, one of the compression standards like H.261 does not have a sequence image start in the data stream, and it does not have picture − and an image in the data stream, either. A start code detector shows a picture end point in an ingress bit stream, and makes a PICTUtZEEND token. It means that the system of this invention carries the data word thoroughly packed so that 1 bit of information might be included in the position of each register selected in order to use it in realization of this invention about this point. 15 bits was selected as the number of bits supplied between two start codes for this purpose. Of course, if it is the usual expert in the technical field concerned, it should be able to understand that it is more than 15 bits, or it is also possible to perform little selection [which]. It is needed for suitable operation that all the data word supplied to a DRAM interface from a start code detector will be 15 bits if it puts in another way. Therefore, a start code detector makes the excessive bit called padding by which the last word insertion of the data token is carried out. 15 data bits were selected for explanation.
[0363]The binary 0 which many binaries 1 follow in order to perform padding (padding) operation based on this invention is automatically inserted, in order to complete 15 bit data word. Next, this data is supplied via the coded data buffer, and is supplied to the Huffman decoder which removes padding. Therefore, arbitrary numbers of bits can supply the buffer of the size and width which were fixed.
[0364]In one example, in order to identify the slice of a picture, a slice start control token is used. A slice start control token is used in order to divide a picture into a still smaller field. With an encoder, the size of this field is selected and and a start code detector, In order to divide a receiving picture into a still smaller field from a start code detector to the machine subordinate State stage located downstream, this only pattern of a slice start code is identified. The size of a field is chosen by the encoder

used by a re-combination circuit and the control token, in order to be identified with a start code detector and to elongate the coded picture. A slice start code is used mainly for error recovery.

[0365]A start code provides the only method of beginning a decoder, and is continuously explained still in detail about this. If it arranges before the data buffer which had the start code detector coded, as compared with the case where a start code detector is arranged, many advantages will be acquired after a coding data buffer and in front of the Huffman decoder and an animation d demultiplexer. When a start code detector is installed in front of the 1st buffer, 1) an assembly of a token being possible, decryption of 2, for example, a standard control signal like a start code, being possible, the pad of a bit stream being possible before three data goes into a buffer, and, 4) In order to empty a buffer, it is possible to make the suitable sequence of a control token and to put available data into the Huffman decoder from a buffer.

[0366]Most control tokens outputted by a start code detector reflect the syntax element of various pictures and a dynamic image code-ized standard directly. A start code detector changes a syntax element into a control token. In addition to these natural tokens, some are unique and/or a machine subordinate token is generated. It is designed especially in order to use it by the system of this invention, and it is internally and externally unique to a unique token, and the token used in order to assist about the multiplex standard nature of this invention is contained in it. A picture end and a coding standard are contained in the example of this kind of unique token.

[0367]A token is introduced in order to remove some of syntactic differences between encoding specification, and in order to carry out a cooperation operation with error condition. Automatic token generation is performed after sequential analysis of standard subordinate data. Therefore, a space decoder answers equally the token by which direct supply was carried out to the input of the space decoder, i.e., SCD, and the token generated following detection of the start code in the coded data. In order to control the multiplex standard nature of this invention, a series of extra tokens are inserted in 2 line interface.

[0368]MPEG and an H.26l. encoded moving image stream decide to call one of the patterns of these a start image and/or standard subordinate code from now on including a standard subordination non-data identification possible bit pattern. In JPEG, the same function is provided in marker code. These /marker codes to start identify the important portion of the syntax of the coded data stream. The analysis of a start / marker code conducted by a start code detector is a stage of the beginning of the coded data analysis.

[0369]A start / marker code pattern is designed identify a pattern, without decrypting the whole bit stream. Therefore, those patterns can be used based on this invention, in order to give error recovery and a decoder start. A start code detector detects the error in the coded data configuration, and provides the function for giving the start of

a decoder. The error detection capability of a start code detector is explained in detail by next along with the start process of a decoder.
[0370]The aforementioned description was mainly explanation about the relation between the characteristic of a machine subordinate bit stream, and the addressing characteristic of this invention. The next description is explanation of the characteristic of the bit stream of the standard subordinate coded data about a start code detector.
[0371]The peculiar start code configuration selected in order to identify specific compression specification, or an image is used for each standard compression encoding system. Each start code has start code value. Start code value is used in order to identify the type of the operation relevant to a start code [in the language of the standard concerned]. In the multiplex standard decoder of this invention, as already stated, the compatibility is based on a control token and a data token configuration. Into each state machine, circuit formation of the index signal including a flag signal is carried out, and suppose that it is explained suitably below.
[0372]The control for which the start included in the standard concerned and/or a marker code, and other standard words contrastive with data word are used in the machine concerned, and/. Or in order to call off confusion with use of a code and/or a machine subordinate code about the contents of the data token, it is not rare to be identified as an image. Similarly, the term called start code is used in many cases as a comprehensive term on a JPEG marker code, MPEG, and H.261. A marker code and a start are useful for the same purpose. The term called ″flash plate″ (flush) is used for both meanings as the case where a ** flash plate token is meant, and a verb, like [in the case (a ″finishing / a flash plate /″ signal is included) of carrying out the flash plate of the start code detector shift register], for example. In order to call off confusion, a flash plate token is certainly written with a capital letter. All the use of everything but a term (a verb or a noun) is based on a small letter.
[0373]A coded [standard subordinate] input picture input stream includes the changing data and the start image of length. A start image carries the value which tells which operation should be carried out to the data concerned which follows immediately according to a standard together. However, in the multiplex standard pipeline processing system of this invention for which compatibility is needed to a multiplex standard, the system is optimized in order to treat all the functions in all the standards. Therefore, on many conditions a unique start control token, Only about the value contained in the value of the coded signal standard image, it is not only compatible, but, According to the display with the parameter specified to each standard as everyone knows in the technical field concerned, in order to emulate the operation of a standard, you must be controllable in various stages. This kind of all standards are included in this specification as reference data, and are ending.
[0374]It is important to understand the relation between the tokens which emulate

the non-data information which is independent or is included in a standard bit stream by combining with other control tokens. An individual set of an index signal including a flag signal is generated by each state machine in order to treat some processings in the state machine concerned. The value carried by the standard can be used in order to access the subordinate control signal of a machine so that the treatment of standard data and a non-data signal may be emulated. For example, a slice start token is inputted into 2 line interface, as it is a 2-word token, therefore already being stated.
[0375]Even if the data input to the system of this invention is a data source from all the suitable data sources that supply 8-bit data to the 1st functional stage in the start code detectors 51 (drawing 18), such as a disk and a tape, it does not interfere, for example. A start code detector has three shift registers, namely, the 1st shift register is 8 bit width.
The next is 24 bit width, and the next is 15 bit width.
Each register is a part of 2 line interface. The data from a data source is loaded to the 1st register as one single octet-bit byte during one timing cycle period. After that, 1 bit of the contents of the 1st shift register are shifted at a time to a decoding (the 2nd) shift register at a stretch. 24 bit registers come after 24 cycles. An octet-bit byte is loaded to the 1st shift register every 8 cycles. Each byte is loaded to the value shift register 221 (drawing 29), and eight additional cycles empty a value shift register, and since the shift register 231 is loaded, it is used. In order to empty the shift register 231, eight cycles are used, therefore there are still 3 bytes after such 3 operations or 24 cycles in 24 bit registers. The value decoding shift register 230 is still empty.
[0376]If it assumes that it has one picture start word in 24 bit shift registers now, a detection cycle will recognize a picture start code pattern, and will supply a start signal as the output. Once a detector detects a start, the byte following it will be a value relevant to the start code concerned.
And this is among sitting (sitting) at the present value register 221.

[0377]Since the contents of the detection shift register are ending with discernment as it is a start code, in order to guarantee that processing does not break out any more, these 3 bytes must be used and these contents must be removed from 2 line interface. A decoding register is emptied and the value decoding shift register 230 waits to shift the value concerned even to this seed register.
[0378]In this stage, the contents of the low-bit position of a value decoding shift register contain the value relevant to a picture start. A space decoder equivalent to a standard picture start signal is SD. It is called a picture start signal. In here, it is SD. The picture start signal itself is going to be included in the token header, and this value is going to be contained in the extended word over a token header.
[0379]10. A token is an omnipotent adaptation unit of the form of an interactive

interface messenger cable package of as opposed to control and/or a data function when realizing token this invention.
And adaptation in order to use it with the processing stage (RPS) which is one stage which reconstructs itself in order to answer one recognized token and to perform various operations and which can be reconstructed.
In order to attain various functions, even if a token is either position subordination or location independence, it does not interfere according to a processing stage. Since a pipeline is supplied by the downstream direction in order to be changed by the processing stage and to attain another function to the next, even if a token is deterioration nature, it does not interfere. A token has a dialog with portions fewer than all or all of stages, and ********** which will have a dialog with the stage of contiguity and/or not adjoining if it carries out from this viewpoint does not interfere, either. A token is position subordination to some functions.
And to other functions, even if it is location independence, it cannot interfere, and the specific dialog with a certain stage can be adjusted with the processing history before the stage concerned.

[0380]Picture - and a token are the methods of transmitting a picture end in a multiplex standard decoder.
[0381]A multiplex standard token is the method of mapping MPEG, JPEG, and an H.261 data stream in one single decoder using the mixture of standard subordination and standard independent hardware, and a control token.
[0382]A search mode token is the art for searching MPEG and JPEG which make possible random access and strengthened error recovery, and an H.261 data stream.
[0383]A stop after picture token is the method of attaining the clear end of decryption. The end of a picture is transmitted and change of a decoder pipeline, i.e., a channel, is cleared.

[0384]Carrying out the pad of the token is the method of making arbitrary numbers of bits supplying through the buffer of fixed size fixed width.
[0385]This invention is turned to the pipeline processing system which has the variable composition which uses a token and 2 line system. Use of the control token and data token which were combined with 2 line system makes easy achievement of the multiplex standard system which makes it possible to have the operation capability extended as compared with the system not using a control token.
[0386]A control token emulates the operation of the standard subordinate signal of a type that a large number supplied in an in-series pipeline processor differ, in order to be a circuit in a decoder processor, and to be generated and to deal with it. The art of examining all the parameters of the multiplex standard selected in order to perform consideration about the item shown in the processing and the next by a sequential

processor is used. 1 [namely,] -- the similarity of them, and 2 -- a difference of them, and 3 -- selection of the right token function for processing effectively all the standard signals sent into the necessity, necessary condition, and 4 sequential processors of them. The function of a token is emulating a standard. A control token function is selectively used as the emulation / translation between standard subordinate signals, and an element for giving control information via a pipeline processor.
[0387]In the system by precedence this art, a standard is identified, and in order to make a dedicated communication circuit by using a microprocessor interface, an exclusive machine is designed according to well-known art. The signal from a microprocessor is used in order to control data flow via an exclusive downstream component. Selection of this extension function, timing, and systematization are performed under control by the fixed logic circuit helped by the signal which carries out ingress from a microprocessor.
[0388]In contrast with this, the system of this invention constitutes a downstream functional stage under control of a control token. It is required from MPU, and/or an option is provided in order for substitution control to come to hand.
[0389]A token provides and creates the format which excelled for transmitting information via an expansion circuit pipeline processor. It is selected in the desirable example shown below, and in the design used, the width of each word of a token is 8 bits of minimum, and one single token can be extended exceeding one or more words. The width of a token is variable.
And the number of bits can be selected arbitrarily.
An extension bit shows whether the bit followed and kicked to all the words of tokens other than the last word of whether a token is extended exceeding the present word and a token is set to the binary 1. When the 1st word of a token has an extension bit of zero, this shows that the length of a token is 1 word slightly.
[0390]Each token is identified by the address field started in the word [1st] bit 7 of a token. Length is variable and an address field can be potentially extended exceeding two or more words. In a desirable example, the length of an address is less than 8 bits. However, this means the size of the processing process instead of the limited condition in this invention selected so that it might be attained by use of these tokens. The extension bit in the words 1 and 2 is one 1, and note that meaning succession of an addition word is shown by an extension bit identification label. The extension bit in the word 3 is zero, therefore shows the end of the token concerned.
[0391]A token is variable bit length. For example, an extension bit is added to a 9-bit token word, and a total of 10 bits is used. In the design of this invention, the width of an output bus is variable. The width of the output from a space decoder is 9 bits, or appears in a *********** rare ** case 10 bits. In a desirable example, the only token using these excessive bits is a data token, and all other tokens disregard this extra bit.

This should understand that not a limited condition but it is only an example.
[0392]By using a data token and a control token configuration, it is possible to change the length of the data expressed by the number of the bits in one word currently carried by these data tokens. For example, the data bit in the word of a data token, When accessing the random access memory used through this whole in-series extension processor, in order to use it, an address's (11 bits' or 10 bits') being formed combining the data bit in other words of the same data token is ending with examination. It makes it easy to have flexibility [the wide range], and variability is added by this.
[0393]As already stated, a data token carries data to the next processing stage from one processing stage. Therefore, when the characteristic of this token passes a decoder, it changes. For example, in the input to a space decoder, a data token carries the bit serial coded dynamic image data packed to 8 bit word. Here, there is no restriction in the length of each token. However, in order to illustrate the flexibility of this mode of this invention, (setting to the output of a space decoder circuit) and each data token shall have 64 words exactly, and each word shall be 9 bit width. More particularly, a standard encoded signal makes it possible to code the intensity and details from which a picture differs to the message from which length differs. Since a group's 1st picture needs to provide the processing unit most information, it has usually longest number of data bits. As a result, the 1st picture can start extension of as many information as possible. Since the word which follows includes the signal which is different when it compares with the 1st word about the 2nd position in the scanning information field, generally its length is short. According to the request of a standard coding system, a word is made scattered by mutual so that the data of variable quantity may be supplied to the input of a space decoder. However, after a space decoder functions, information is supplied to the output in a picture format rate suitable for the display on a screen. For example, in order to continue and interface with NTSC, PAL, and the various display systems and whole world like SECAM, the output rate about the time of a space decoder can change. An animation formatting part changes this variable picture rate into a fixed picture rate suitable for a display. However, picture data is still carried by the data token which comprises 64 words.
[0394]11. One DRAM interface which can be constituted single highly efficient is used for each of the decoder chip of three DRAM interfaces. Generally, the DRAM interface on each chip is substantially the same. However, an interface differs in the method of treating a channel right of priority mutually. This interface is designed drive directly a space decoder, a time decoder, and external DRAM used by the animation formatting part. Generally, in order to connect a DRAM interface to DRAM of these systems, external logic, a buffer, or a component is not required.
[0395]Based on this invention, an interface can be constituted by the following two methods.

1. The interface of detailed timing can be constituted so that a variously different DRAM type may be accommodated.
[0396]2. The width of the data interface to DRAM can be constituted so that cost / performance trade-off may be provided according to a different use.
[0397]Generally, a DRAM interface is the block which became independent of the standard in a system respectively realized on three chips. Again, these chips are a space decoder, a time decoder, and an animation formatting part. Supposing that drawing 18, drawing 19, and drawing 20 are referred to again, these figures are block diagrams which express the relation between the blocks of the remainder of a space decoder, a time decoder, and an animation formatting part as a DRAM interface, respectively. In each chip, a DRAM interface connects a chip to external DRAM. Since it is not practical to create comparatively a lot of DRAMs needed on a chip now, external DRAM is used. (Notes: Each chip has own external DRAM and an own DRAM interface.) Further, when a DRAM interface is compression standard independence, An interface must be constituted so that each of multiplex standard H.261, JPEG, and MPEG may be realized. It is explained still in detail later how the DRAM interface as an object for multiplex standard operations is reconstructed.
[0398]In order to understand the operation of a DRAM interface, it is required to understand how it communicates among these both using the relation and 2 line interface between a DRAM interface and an address generator.
[0399]Generally, in order to carry out the address of the DRAM, an address generator generates the address which a DRAM interface needs, so that a name may mean (for example, when writing to the specific ** address in DRAM). the case where a DRAM interface has both data and an effective address when using 2 line interface (from an address generator) (from the precedence stage in a pipeline) -- as long as -- reading and writing break out. Use of an individual address generator simplifies the composition of both an address generator and a DRAM interface so that it may inquire further later.
[0400]In this invention, it is operational in a DRAM interface from the clock which synchronizes with both an address generator and the clock of a stage with which data is supplied via it. Special art is needed in order to treat the asynchrony of operation.
[0401]Generally data is transmitted between a DRAM interface and the remaining portion of the chip in 64 bytes of block (only the prediction data in a time decoder is the only exception). Transmission is performed by the device known as a "swing buffer." or [that a swing buffer fills one RAM intrinsically] -- or, using empty -- simultaneous -- while -- **** -- they are one pair of RAM operated in the configuration by which the double buffer was carried out using the DRAM interface which other parts of a chip empty another rum, or fills. The individual bus which carries the address from an address generator is connected with each swing buffer.
[0402]In this invention, although each chip has four swing buffers, the functions of

these swing buffers differ for every case. In a space decoder, one swing buffer, Use the coded data for DRAM at a ****** sake, the swing buffer of one ** is used in order to read the data coded from DRAM, and the 3rd swing buffer, It is used in order to transmit to DRAM the data by which tokenization was carried out, and it is used in order that the 4th swing buffer may read from DRAM the data by which tokenization was carried out. However, in a time decoder one swing buffer, It is used in order to write intra or prediction picture data in DRAM, the 2nd swing buffer is used in order to read INRA or prediction data from DRAM, and it is used in order for other two to read the prediction data to the front and back. In an animation formatting part, one swing buffer, being used in order to transmit data to DRAM, and being used in order for other three to read data from DRAM -- luminosity (Y), red, and blue color difference data (each, Cr and Cb) -- one is used at a time as business. Suppose that the operation of one virtual DRAM interface of having one write-in swing buffer and one reading swing buffer is explained in the next section. Intrinsically, this is the same as the operation of the DRAM interface of a space decoder. Operation is shown in drawing 32.
[0403]It is shown that all the control interfaces between the remaining stages of the chip which drawing 32 interfaces [address generator 301 and DRAM] 302, and supplies data are 2 line interfaces. Even if the address generator 301 generates the sequence which generated the address as a result which receives a control token, or the address only fixed (for example, for the FIFO buffers of a space decoder), it does not interfere. A DRAM interface treats 2 line interface connected with the address generator 301 by a special method. Instead of holding acceptance to a high, when the preparation whose acceptance line receives an address is completed, in the effective address supply by an address generator, an address line processes waiting and the address concerned, and covers 1 clock period, and sets an acceptance line to a high. Thus, a request / positive-acknowledge (PEQ/ACK) protocol is realized.
[0404]The peculiar feature of the DRAM interface 302 is the ability to be able to supply the address generator 301 and data, or to communicate with the stage to accept independently. For example, even if an address generator generates the address connected with the data in a write-in swing buffer (drawing 33), it does not interfere, but. However, action is not performed until it transmits that the block of the data in which the entry preparation for external DRAM was completed by the write-in swing buffer exists. Similarly, although a write-in swing buffer includes the block of the thing data with which the entry preparation for external DRAM is complete, it does not interfere, but action is not performed until an address is supplied to the bus concerned from the address generator 301. Once one of the RAM in a write-in swing buffer is filled with data, Before ****(ing) data input (2 line interface acceptance signal is set to a low), other RAM is filled thoroughly and "may be swung" at the DRAM interface side.
[0405]When you understand the operation of the DRAM interface 302 concerning this

invention, in the system constituted appropriately a DRAM interface, Between a swing buffer and external DRAM303, it is important that it is cautious of data transfer being possible by a data rate equivalent to the sum total of all the average data rates between a swing buffer and the remaining portion of a chip at least.

[0406]Each DRAM interface 302 determines which swing buffer gives one's service to the next. Generally, this is "round-robin" (round robin) (that is, the swing buffer which receives service next). or it is the following available swing buffer which was the turn which does not come out most these days, at least priority is either of the encoders (that is, some swing buffers have at least priority still higher than other things in this case). Ingress of the additional request which has at least priority still higher than other requests of all the in the case of both sides is carried out from a RIFURRESHU request generator. A refresh request is generated from a programmable RIFURESHU counter via a microprocessor interface.

[0407]Here, it supposes that drawing 33 is referred to and the lineblock diagram of a write-in swing buffer is shown in this figure. Two blocks, the 1RAM311 and 2nd RAM312, are included in a write-in swing buffer interface. Data is entered in RAM331 and RAM312 from a front stage under control by the write-in address 303 and the control section 314 so that it may inquire further here. Data is entered in DRAM315 from RAM311 and RAM312. A DRAM line address is supplied by an address generator, and a column address is supplied by a write-in address and the control section so that it may be further explained during the data entry to DRAM here. In operation, effective data is supplied to the input 316 (data yne). Generally, data is received from a front stage. RAM311 is filled in, and a write-in address control section **************s RAM311, and enables entry to RAM311 of the following one data as each data is received by a DRAM interface. The entry to RAM311 of data is continued until there is no data more or RAM311 fills. When RAM311 is full, an input side gives up control, and in order to show that RAM311 is [reading] ready, it sends a signal to the reading side. Therefore, this signal supplies between two asynchronous clock regimes, and supplies it via three synchronization flip-flops. If RAM312 assumes that it is empty, the next item of the data which reaches an input side will be written down in RAM312. When that is not right, and RAM312 becomes empty, this occurs. When round-robin ** points to the swing buffer with which at least priority hits the turn that an encoder is read (which is used by a specific chip, and responding), a DRAM interface reads the contents of RAM311 and enters them in external DRAM. Next, a signal is back sent through an asynchronous interface and shows that the preparation again made by RAM311 is complete.

[0408]When a DRAM interface empties RAM311, and "swinging" it before an input side fills RAM312 and, the continuation acceptance by the swing buffer of data is possible. When that is not right and RAM2 is full, a swing buffer sets the acceptance signal to a low until RAM311 "is swung" back for use by an input side.

[0409]The operation of a reading swing buffer based on this invention is the same if the point which input data and output data reverse is removed.
[0410]The DRAM interface of this invention is designed make an available memory bandwidth into the maximum. Eight xeach 8 blocks of data are memorized by the same DRAM page. Thus, DRAM high-speed page access mode can be used thoroughly. In this case, one line address is supplied and many column addresses follow this. Although especially a line address is supplied by an address generator, a column address is supplied by a DRAM interface so that it may inquire later. The function which makes it possible that the width of the data bus to external DRAM is 8, 16, or 32 bits is provided. Therefore, the amount of DRAM used can be made to match the size and the bandwidth necessary condition of a particular application.
[0411]In this example (say correctly how the DRAM interface on a space decoder operates), an address generator is read and supplies the block address to each of a write-in swing buffer to a DRAM interface. This address is used as a line address for DRAM. A 6-bit column address is supplied by the DRAM interface itself, and these bits are similarly used as an address for swing buffer rum. The width of the data bus to a swing buffer is 32 bits. Therefore, when the bus width to external DRAM is less than 32 bits. DRAM access of the exterior of 2 or 4 must be performed, before reading the following word in a write-in swing buffer, or before the following word's reading and filling in a swing buffer (reading [and], writing means the direction of the transmission to external DRAM).
[0412]In the case of a time decoder and an animation formatting part, conditions are still more complicated. Addressing of a time decoder is still more complicated because of that prediction mode so that this section may be examined further. Addressing of animation formatting for a multiplex animation output specification mode is still more complicated so that this section may be further examined about an animation formatting part.
[0413]As already stated, a time decoder has four swing buffers. Two of them are used in order to perform reading and the writing of decrypted Yingde and prediction (I call P) picture data. These operate, as already explained. Another two are used in order to receive prediction data. These buffers are still more interesting.
[0414]Generally, prediction data inclines from the position of a processed block as specified as the movement motion vector as x and y. Therefore, the data block which should be searched is that (and written in DRAM) which was coded, and, generally is not in agreement with the block border of the data concerned. The outline which the portion to which this attached shade as shown in drawing 34 expresses the block formed, and is shown by a dotted line expresses BURROKKU by which prediction is being made based on it. An address generator changes the address specified by the motion vector into the pixel offset so that it might change into the offset block (total of a block) and a small arrow might show, as a large arrow shows.

[0415]In an address generator, in order to form the address of the block which should be searched from DRAM, a frame pointer, a base-block address, and vector offset are added. When pixel offset is zero, only one request is generated. When either x or y dimension has offset, two requests, i.e., the block address of a basis, and a block address just below are generated. Four requests are generated when x and y have offset. About each block which should be searched, an address generator is started and calculates a stop address as best shown in the example of the figure.
[0416]Suppose that the pixel offset (1, 1) equivalent to the portion of the shade of drawing 35 is considered. An address generator makes four requests by which a label indication is carried out by D from A in a figure. The problem which should be solved is saying how the necessary sequence of a line address is supplied at high speed. An answer is using ″start/stop″ art, as shown below.
[0417]Suppose that the block A of drawing 35 is considered. Reading must be started in a position (7, 7) and must be ended in a position (1, 1). In a present stage, it is assumed that it is that in which 1 byte is being read at a stretch (namely, 8-bit DRAM interface). x value in a coordinates pair forms three LSB of an address, and y value forms three MSB. Both sides are 1 and x and y start value provide an address with 9. Data is read in this address and the ink lint of the x value is carried out. When a process is repeated until x value reaches the stop value, and it reaches, y value **************s only 1, and x start value is reloaded and gives the one address 17. It **************s x value again until it reaches stop value, as each byte of data is read. A process is repeated until both the value of x and y reaches those stop value. Thus, 9, 10, 11, 12, 13, 14, 15, 17...23, 25, --, 31 and 33, --, the address sequence of 57, --, 63 are generated.
[0418]In the same method, the start to the block B and stop coordinates are (1, 0), and (7, 0), and are (0, 7) as opposed to (0, 1) the block C, and are (0, 0) as opposed to (0, 0) the block D.
[0419]The following problem is where to fill in this data. Clearly, if the block A is observed, the data read in the address 9 must be entered in the address 0 of a swing buffer, and the data from the address 10 must be entered in the address 1 of a swing buffer. ****. Similarly the data read in the address 8 in the block B must be entered in the address 15 in a swing buffer, and the data from the address 16 must be entered in the address 15 in a swing buffer. This function can be realized very easily so that it may next give an outline.
[0420]The block A shall be considered. As for a swing buffer address register, the reverse value of stop value is loaded on the occasion of a reading start. The reverse stop value of y forms three MSB, and the reverse stop value of x forms three LSB. In this case, although a DRAM interface is reading the address 9 in external DRAM, a swing buffer address is zero. Yne KURIREMENTO also of the swing buffer address register is carried out as are shown in Table 500 ″prediction addressing″ and it

**************s an external DRAM address register.

[0421]The explanation so far has been concentrated on the 8-bit DRAM interface. In 16 or a 32-bit interface, trifling correction of 2 and 3 must be made. First, pixel offset vectors "must be clipped" so that it may point out 16 or 32 bit boundaries. In the example used here, about the block A, the 1st read DRAM points out the address 0, and the data to the addresses 0-3 is read. Garbage data must be abandoned by the 2nd. This writes all data in a swing buffer (in this case, it must be physically larger to a difference than that the swing buffer was required in the case of 8 bits), and is carried out by reading with offset. When carrying out half a pixel of MPEG interpolation, 9 bytes in x and/or y must be read in a DRAM interface. In this case, an address generator provides a suitable start and stop address. Although some addition logic is used in a DRAM interface, there is no fundamental change in the way the DRAM interface operates. The point of the last which it should be careful of about the time decoder DRAM interface of this invention is that a prediction filter must be provided with additional information, in order to show whether it is needed what kind of processing to carry out about data. The contents of required processing are shown below.

[0422]The "last byte signal" which shows the byte of the last of transmission (64, 72, or 81 bytes).

[0423]H.261 flag.

[0424]Bidirection prediction flag.

[0425]Two bits which show the dimension (8 or 9 bytes in x and y) of a block.

[0426]One 2 bit numbers which show an order of a block.

[0427]A last byte flag is generable as data is read from a swing buffer. A pipe lets other signals pass through a DRAM interface so that a signal may be combined with the right block of data, as it is obtained from an address generator and data is read from a swing buffer by the prediction filter block.

[0428]In an animation formatting part, data is read in order of a raster, although external DRAM in a block is filled in. Although writing is completely the same as the case where a space decoder is already explained, only few of reading is still more complicated.

[0429]The data in an animation format part and external DRAM is organized so that the data of at least eight blocks may suit one single page. These eight blocks are eight continuous horizontal blocks. It is required to read eight bytes from each of eight continuous blocks, and to fill in a swing buffer when rasterizing (namely, the same sequence in 8-block each).

[0430]It supposes that the best sequence is considered (and the interface of 1 byte width is assumed), and x address (3LSBS) is set to zero like the case of y address (3MSBS). Next, it **************s x address as 8 bytes of the beginning are read respectively. At this time, it **************s the top part article of the upper portion

(the bit 6 and the more than it-LSB= bit 0) of an address, and x address (3LSBS) is reset by zero. This process is repeated until 64 bytes is read. When the interface width to external DRAM is 16 or 32 bits, it only **************s x address only 2 or 4, respectively instead of **************ing only 1.
[0431]In this invention, although 8 bytes of multiple is always read, the address generator can transmit to a DRAM interface that less than 64 bytes must be read (this is needed on the occasion of a start or end of a raster line, and has things). This is attained by using a start and stop value. Stop value is compared with start value in order that start value may generate the signal which is used for the top portions of an address (six or more bits), and shows the time of reading being stopped.
[0432]In order to put the edge of a DRAM signal on 1/4 of the accuracy of the clock period of a system, the DRAM interface timing block in this invention uses a timing chain. Two rectangular clocks obtained from the loop by which phase clocks were carried out are used. These clocks are put together and form 2x clock on a concept. Therefore, all one chains are made with two shift registers with a phase opposite to 2x clock by which multiple connection was carried out.
[0433]First, in the first place, there is a chain as an object for page start cycles, and one chain is read/write/object for refresh cycles now. Via a microprocessor interface, it is programmable, the length of each cycle is set after that, and a page start chain has the fixed length, and the length of a cycle chain changes suitably during the page start.
[0434]If it resets, a chain will be cleared and a pulse will be made. A pulse moves along with a chain and is oriented by the state information from a DRAM interface. A pulse generates a DRAM interface clock. Each DRAM interface clock period is in agreement with one cycle of DRAM. Therefore, when the length of a DRAM cycle differs, a DRAM interface clock is not constant speed.
[0435]An additional timing chain combines the pulse from the above-mentioned chain with the information from a DRAM interface, for example, generates the output strobe and enabling like notcas, notras, notwe, and notbe. 12. In prediction filter re-****19, drawing 20, and drawing 21, the lineblock diagram of a time decoder is shown in drawing 40 still in detail. A prediction filter is contained in this. The relation between a prediction filter and the remaining elements of a time decoder is shown in drawing 26 still in detail. The essence of the structure of a prediction filter is shown in drawing 27 and drawing 37. The detailed description about the operation of a prediction filter is indicated to "a still more detailed description of an invention" of this section.
[0436]Generally, although the prediction filter based on this invention is used in MPEG and H.26l. mode, it is not used in JPEG mode. Please recollect the case in JPEG mode. That is, a time decoder only makes data supply through an animation format part, without performing all substantial decryption exceeding the range attained by the space decoder. Supposing that drawing 27 is referred to again, in an MPEG mode, the

forward and the backward prediction filter are the same, and filter each MPEG forward and a backward prediction block. However, since H.261 does not use backward prediction, only a forward prediction filter is used in H.261 mode.
[0437]Each of two prediction filters of this invention is substantially the same. Suppose that it supposes that drawing 17 and 505 are referred to again, and more particularly drawing 37 is referred to. The block diagram of the structure of a prediction filter is shown in a figure. Each prediction filter has four stages by which serial arrangement was carried out. Data is inputted into the format stage 505-7, and is made the format which may be filtered easily. In the following stage 505-2, I-D prediction is performed on an X coordinate. After required transportation is performed by the dimension buffer stage 505-3, I-D prediction is performed on the Y coordinate in the stage 505-4. A stage decides to explain still in detail about how filtration is carried out. The conditions required of filtration are defined by the compression standard. In the case of a low pass filter, the filtration which is performed actually in the case of H.261 is similar.
[0438]Again, with reference to drawing 26, in order that multiplex standard operation may carry out either MPEG or H.261 filtering, a prediction filter needs to be able to reconstruct or not to carry out filtration at all in JPEG mode. A prediction filter is reconstructed by the token like the mode in which many reconstruction of everything but 3 chip system is possible. A token is used also in order to notify to an address generator about the specific mode of operation. In this case, the address generator can supply the address of the required data which changes remarkably to a prediction filter between MPEG and JPEG.
[0439]13. Most registers in an accessing register microprocessor interface (MPI) can be restricted when the stage to which the interface concerned relates has stopped, and they can be corrected. Therefore, generally the group of a register is connected with the access register. When the value in an access register is zero, the group of the register connected with the specific access register concerned must not correct. If 1 is entered in an access register, it will be required that a stage should be stopped. However, since the stage cannot stop immediately therefore, a stage access register holds value zero until it is stopped.
[0440]In relation to MPI, it must wait for all the user software used in order to carry out a function via MPI until 1 is read in the access register "set after entering 1 in a request access register." When a user enters one value in configuration registers, the result is unknown until it can come, simultaneously the access register is set to zero.
[0441]14. In all the circuits in a microprocessor interface space decoder and a time decoder, a microprocessor interface (MPI) with standard byte width is used. MPI operates in asynchronous with various space and a time decoder clock. With reference to Table 37, various MPI signals used in this interface are shown in this table. The character of a signal is shown in input output fields, and the name of a

signal is shown in the signal name column, and a signal symbol description is shown in the description column. The electric specification of MPI is shown about Table 38. All the specifications are classified according to a type, and three types are shown in the sign column. It is shown in the parameter column what these signs express. Actual specification is shown in each column in the minimum, the maximum, and a unit.
[0442]DC operating condition is shown about Table 39. The title of a column is the same as the case of Table 38. DC electric characteristics are shown about Table 40, and the same **** as the case of Table 38 and 39 is used.
[0443]15. The AC characteristics of an MPI lead timing MPI lead timing diagram are shown about drawing 62. Each line of a figure is carried out by the corresponding signal name with a label, and NANOSEKANDO shows timing. The perfect lead timing characteristics of a microprocessor interface are shown about Table 41. A column number shows the signal corresponding to the signal name written in the characteristic column. The column identified by MIN and MAX provides the minimum length of time to show the peak of the time when the signal concerned is available. An unit section shows the measurement unit used in order to explain a signal.
[0444]16. A general description of an MPI light timing MPI light timing diagram is shown about drawing 63. This figure shows the signal name according to each relevant to MPI light timing. A name, the characteristic of a signal, and other various physical characteristics are shown about Table 42.
[0445]17. In keyhole address location this invention, the memory map with small access frequency is arranged behind the keyhole register. A keyhole register has two registers connected with it. The 1st register is a keyhole address register.
And the 2nd register is a keyhole data register.
A keyhole address specifies the place in an extension address space. Reading or the write-in operation to a keyhole data register accesses the place specified with the keyhole address register. After accessing a keyhole data register, a related keyhole address register is **************ed. The random access in the extended address space is possible only by writing a new value in a keyhole address register to each access. the circuit of this invention within the limits -- two or more keyhole map **** -- there may also be things. Nevertheless, there is no dialog between different keyholes.
[0446]18. It supposes a picture and that drawing 18 is referred to again, and the general lineblock diagram of the space decoder used for this invention is shown in this figure. The description about the function of a picture end is continued and applied to this whole lineblock diagram. Picture - and a function have an advantage in the multiplex standard that H.261 coding picture data, MPEG, and a JPEG signal can be treated.
[0447]As already stated, interconnection of the overall lineblock diagram of drawing 18 is carried out by already explained 2 line interface. Each functional block is

arranged so that it may operate based on the state machine configuration shown about drawing 17.
[0448]Generally, in the start code detector which generates picture − and a control token, the picture end based on this invention is started. Picture − and a control token are supplied to a DRAM interface, while it has not changed through a start-up control circuit. In this case, said token is used in order to carry out the flash plate of the light swing buffer in a DRAM interface. The contents of the swing buffer should recollect restricting, when this buffer fills, and being written in RAM. However, a buffer is completed also in the point which is not full and adherence of picture data may be made to cause. Picture − and a token force data out of a swing buffer.
[0449]Since this invention is a multiplex standard machine, this machine differs and operates to each compression standard. Suppose that the case where it operates according to a machine subordinate action cycle is fully explained more particularly. To each compression standard, the total number which are all the available action cycles can be selected also by the design of the control token itself so that it is possible to select the total number of all the available action cycles with the combination of a control token and/or the output signal from MPU. This invention is organized so that it may make it postpone that information goes into the following block, until all the information will be collected by upstream block about this point. A system stands by until the preparation to which data is made to supply to the following stage is completed. Thus, picture − and a signal are supplied to the coded data buffer, and the control section of picture − and a signal is supplied to the reading length lifting, the Huffman decoder, and animation demultiplexer circuits of the contents of a data buffer.
[0450]Other advantages of picture − and a control token, Even if it is a case where it does not have a total range where the signal supplied to the Huffman decoder and animation demultiplexer circuits is generally predicted, and/or a number about use by the Huffman decoder demultiplexer, it is that this control token identifies the termination of a picture. On this condition, the information currently held at the coded data buffer is supplied to the Huffman decoder and an animation demultiplexer as a perfect picture. Thus, the state machine of the Huffman decoder and an animation demultiplexer can still treat data according to a system design.
[0451]Other advantages of picture − and a control token are the capability to empty a coding data buffer thoroughly, as floating information does not remain carelessly to off-chip DRAM or a swing buffer.
[0452]The advantage of further others of picture − and a function is the use in error recovery. For example, if it assumes that they are things fewer than the quantity generally used in order to show spacial information about the single image whose quantity of the data currently held at the coding data buffer is one, the last picture will be held at a data buffer until a swing buffer fills, but. However, according to the

definition, a buffer never fills. In a certain point, a machine determines that error condition leaves. Therefore, the last picture can be decrypted within limits forced so that picture - and a token may be decrypted and the data in a coding data buffer may be supplied to the Huffman decoder and an animation demultiplexer, and information is made them from a buffer in the sky. As a result, a machine does not go into error recovery mode, but improves [the result] processing of the coded data, and continues it.
[0453]Another advantage of the use of picture - and a token is that an in-series pipeline processor continues processing of interruption-less data. By using picture - and a token, an in-series pipeline processor is constituted so that the data of less than the predicted quantity may be treated, and it continues processing as the result. Generally, the machine by the advanced technology originates in error condition, and stops. As already stated, it counts these as a macro block carries out ingress of the coding data buffer to the storage area. The Huffman decoder and the animation demultiplexer know the amount of information generally predicted when decrypting each picture, namely, the state machine portions of Huffman decoding and an animation demultiplexer know the block count processed during each image recovery cycle period. When the block of the number of the rights does not arrive from a coding data buffer, generally an error recovery routine is materialized as a result. However, by using picture - and the control token which reconstruct the Huffman decoder and an animation demultiplexer this control token, Since it tells treating the information on an actually suitable quantity for the Monday coda and an animation demultiplexer by reconstruction, it can continue functioning.
[0454]Supposing that drawing 17 is referred to again, a buffer manager's token decoder portion detects picture - and the control token which were generated by the start code detector. In a normal system operation state, as already stated, about the normal operation of a swing buffer, a buffer register fills and is emptied. The swing buffer again filled with data selectively does not become empty until it gets to know time to become empty until a swing buffer is filled thoroughly and/or. Picture - and a control token force making into a coding data buffer in the sky the swing buffer which was decrypted in a buffer manager's token decoder portion, and has been filled selectively. This is ultimately supplied to the Huffman decoder and an animation demultiplexer via a direct or DRAM interface.
[0455]19. Other advantages of flashing operation picture - and a control token are the function relevant to a flash plate token. A flash plate token is connected with neither reconstruction control of a state machine, nor the data path to a system. This token, not to mention it, completes a front partial signal, in order to treat with the state machine of machine subordination. Each state machine is processed in a flash plate control token, and a hand is recognized as information not becoming. Therefore, a flash plate token is used in order to fill all the parts of the remaining empty of a coding

data buffer, and it is used in order to make it possible to send sufficient set of information to the Huffman decoder and an animation demultiplexer. Thus, a flash plate token acts on Mr. Padding to a buffer.
[0456]The token decoder in the Huffman circuit recognizes a flash plate token, and disregards the data of the falsehood which the flash plate token stuffed into it. Next, the Huffman decoder acts only on the data content of the last picture buffer, when it exists before arrival of picture –, a token, and a flash plate token. It is picture – and token independent, or the further advantages of the use combined with the flash plate token are reconstruction of the Huffman decoder circuit, and/or reorganization. It gets to know that the Huffman decoder circuit has information less than the information usually predicted in order to decrypt the last picture on the occasion of arrival of picture – and a token. the Huffman decode circuit ending processing of the information included in the last picture, and passing a DRAM interface for this information –– a reverse modeler –– it outputs. When identifying the last picture, the Huffman decoder becomes cleanup mode and is readjusted to arrival of the following picture information.
[0457]20. A buffer in order to supply a flash function flash plate token through the whole pipeline processor based on this invention is used in order to guarantee that it is empty and being reconstructed so that it may wait for arrival of data with other new circuits. More particularly, this invention has a flash plate token which directs that picture – and the combination of a token, a padding word, and image processing to the form of the present picture are completed to an in-series pipeline processor. It sets after that, and various state machines need to reconstruct so that it may wait for arrival of new data, in order to treat newly. A flash plate token should be careful of acting as special reset to a system. A flash plate token resets each stage when passing. However, continuation of processing of the following stage is enabled. This prevents a loss of data. If it puts in another way, a flash plate token will be variable reset in contrast with absolute reset.
[0458]21. A stop after picture stop after picture facility is used in order to close an in-series pipeline's expansion circuit in the logical point in the operation. Picture – and a token are generated at this time, and data shows that completion and padding operation completed the ingress from a data input line. A padding function fills an empty data token selectively. Next, a flash plate token is generated, and is supplied through an in-series pipe-line system, and extrudes all the information from a register, and forces that a register returns to those neutral standby states. Next, a stop after picture event is generated, and an input is not accepted until either a user or a system clears this state. If it puts in another way, picture – and a token will transmit the end of a picture, and stop after picture operation will transmit simultaneously that all the present processings finished.
[0459]22. Other features of multiplex standard search mode this invention are use of

the SEARCHNODE control token used since the input to an in-series pipeline processor is reconstructed so that the bit stream which carries out ingress may be investigated. When search mode is set, a start code detector looks only for a specific start code or the marker used for one with an arbitrary compression standard. However, please understand that other images of other data bit streams can be used for this purpose. Therefore, these images cover whole this invention, and they can be used in order to change into other examples which can use a data token with a reconstruction circuit in order to provide the combination of a control token, and the same processing.
[0460]On many conditions to include, use of the search mode in this invention the case where it is shown below, [convenient] Namely, when cutting of 1 data-bit stream breaks out, two users change a channel intentionally and a data bit stream is cut, For example, when based on user operation-ization of the high-speed forward direction from a controllable data source like arrival of the data based on a cable conveyance compression digital animation, 3, for example, an optical disk, or an animation disk, or an opposite direction. Generally, search mode is convenient when a user interrupts an in-series pipeline's normal processing in the point that a machine does not predict interruption.
[0461]When either of the search modes is set, a start code detector looks for the start image for which it was suitable in order to make a machine-independent token and which carries out ingress. All the data which carries out ingress to a start code detector before discernment of a standard subordinate start image is meaningless, when the machine is waiting for this information, by an idling condition, the back carries out a machine and it is abandoned.
[0462]The start code detector can assume one with arbitrary numerical 0 configuration composition. For example, one of the configurations of these enables search about the group or the higher-ranking start code of a picture. With this pattern, a start code detector abandons all those inputs, and looks for a group start standard image. When this kind of image is identified, a start code detector generates a group start token, and search mode is reset automatically.
[0463]It is important for one single circuit, the Huffman decoder, and an animation demultiplexing circuit that it is cautious of operating with the combination of an input signal including a standard independent setup signal and a coding standard signal. A coding standard signal is transmitting information directly from the bit stream which carries out ingress according to the request by the Huffman decoder and an animation demultiplexing circuit. However, on the other hand, the Huffman decoder and an animation demultiplexing circuit function under the operation of a standard independent sequence signal.
[0464]This operation mode was the most effective, and since it designed instead of carrying the actual signal itself when carrying a standard subordinate input to the

Huffman decoder and an animation demultiplexer so that a special control token might be used, this operation mode was selected.
[0465]23. Reverse modeler inverse model-ization is the feature of these three standard ******, and is the same to all three standards. Generally, the data token in token buffers includes the information about the value of the quantized coefficient, and the information about the number of the zero between the coefficients displayed (one form of run length coding). The reverse modeler of this invention is adapted so that it may be used with a token.
And the information about execution of zero is simply expanded so that each data token may contain 64 value as an indispensable condition.
The value in a data token which can be set after that is the quantized coefficient which can be used by an inverse quantization device.
[0466]24. In the way the whole IC set can treat multiplex standard data, the inverse quantization device of inverse quantization device this invention was realized, although it was an element needed in a decryption sequence. An inverse quantization device is ending with adaptation so that it can be used with a token. An inverse quantization device is arranged between a reverse modeler and reverse DCT (IDCT). For example, in this invention, before data moves to IDOT, the adding machine in an inverse quantization device is used in order to apply a constant to the number of pixel decodings.
[0467]The number of pixel decryption which changes according to each standard used in order to code information is used for IDOT. In order to decrypt information correctly, before data continues entering IDCT, the value 1024 is added to the number of decodings by the inverse quantization device.
[0468]In order to treat the data compressed by various standards by already using the adding machine which carries out the whereabouts in an inverse quantization device in order to standardize data, before data reaches IDCT, the necessity of adding a circuit or software to IC is eliminated. Suppose that the operation besides preparation ****** for multiplex standard operations is inquired in the period middle of an "after-quantization function" in a line crack and the back.
[0469]The control token which accompanies data is decrypted and, below, various standardization routines with required being carried out by the inverse quantization device are identified in detail. All the functions "after these quantization" are realized so that duplication of a circuit may be avoided and IC can treat multiplex standard decoding data.
[0470]25. The Huffman decoder and a parser (parser)
Supposing that drawing 18 and drawing 36 are referred to again, a space decoder has the Huffman decoder for decrypting the data Huffman-code-ized according to various compression standards. Although each of the standards JPBG and MPEG and H.261 requires that Huffman encoding of the specific data should be carried out, in one side,

a difference of the Huffman decrypting method which each standard requires has an important meaning. In the space decoder of this invention, this invention saves a precious die space by creating only these common modes only once by identifying the common mode of each Huffman decoder, without designing and creating every a total of three one individual Huffman decoders to each standard. A wise multiplex section article algorithm, i.e., an algorithm which strengthens further the similarity of the mode of each Huffman decoder which has similarity to other standards, is used. If it summarizes, the Huffman decoder 321 will operate with other units shown in drawing 36. Other units of these are the parser state machine 322, the yne shifter 323, the data index unit 324, ALU325, and the token formatting part 326. As already stated, the connection between these blocks is managed by 2 line interface. It supposes that it is described continuously how these units function still in detail, and here doubles and explains a focus to the specific mode which supports the multiplex standard operation of the Huffman decoder based on this invention.
[0471]The parser state machine of this invention is a programmable state machine which acts so that OPERESHO of other blocks of an animation parser may be unified. A parser state machine controls other blocks on which this control word acts by generating the control word which answers data, and parallel arrangement is carried out to data, and is supplied to other blocks. Being not only useful but it is indispensable to juxtapose with associated data and to supply control word, since a block is connected via 2 line interface. Thus, data and control arrive simultaneously. By a control line, passage of control word passes along the bottom of the data line which connects a block, as shown in drawing 36. Especially this code word identifies the specific standard currently decrypted.
[0472]The Huffman decoder also carries out a certain kind of control facility. Especially the Huffman decoder 321 contains the state machine which can control the specific function of data index part 324 and ALU325. These units by the Huffman decoder need to be controlled in order to decrypt block level information correctly. If these decision is made to make to the parser state machine 322, it will take time too much.
[0473]The important mode of the Huffman decoder of this invention is the capability to reverse the data bit coded when reading a data bit to the Huffman decoder. This capability is needed in order to decrypt an H.261 Huffman style code. The Huffman code of the particular type with which a reason is used by H.261 (substantially MPEG) is because it has polarity opposite to the code used by JPEG. Therefore, the same table as the table which the Huffman decoder uses is substantially made usable to all three standards by using an inverter. In the section of ″still more detailed explanation of an invention″, other modes of the way the Huffman decoder realizes all three standards are explained still in detail.
[0474]The data index unit 324 carries out the 2nd portion of a multiplex section article

algorithm. This unit contains the look-up table (table for reference) which provides the actual Huffman decoding data. The entry to this table is organized based on the number of indexes generated by the Huffman decoder.

[0475]ALU325 realizes the remaining portion of a multiplex section article algorithm. Especially ALU treats sign extension. ALU contains the register file holding vector prediction and DC prediction. In the section related to a prediction filter, use is described to this file. ALU contains the counter counted via the structure of the picture currently decrypted by the space decoder. Especially the dimension of a picture is programmed by the register relevant to a counter, and makes easy detection of a "picture start" and a macro block code start.

[0476]Based on this invention, the token matching part 326 (TF) assembles the decrypted data to a data token. These tokens are supplied to the remaining stages or block in the decoder between space-time next.

[0477]In this invention, a yne shifter receives data from FIFO which buffers the data which supplies a start code detector. That is [generally there are two types of data which a yne shifter receives], as the section of a "token" is examined further, they are a data token and a start code exchanged for each token by a start code detector. Most data should be careful of being a data token which needs decryption.

[0478]A yne shifter supplies data to the Huffman decoder in in-series.

On the other hand, a control token is supplied in parallel.

In the Huffman decoder, the data by which Huffman encoding was carried out is decrypted according to the 1st portion of a multiplex section article algorithm. Especially a specific Huffman code is identified, next is exchanged for the number of indexes.

[0479]The Huffman decoder identifies the specific data which needs the special treatment by other blocks shown in drawing 36. This data includes a block end and an escape. In this invention, there is no time then in a data index unit, and it is saved by detecting these in the Huffman decoder. Next, this number of indexes is supplied to the data index unit 324. Intrinsically, a data index unit is a look-up table. The RUKKO rise table based on one mode of an algorithm does not have the Huffman code table specified by JPEG and great difference. The format which it generally specifies in order that JPEG may transmit an alternative JPEG table is a compression data format.

[0480]The decrypted number of indexes or other data are supplied to ALU 504-5 which carries out operation already described with accompanying control word from the data index unit 324.

[0481]Data and control word are supplied to the token format part 326 (TF) from ALU325. In a token format part, data is combined with control word, in order to form a token if needed. Next, a token is carried by the next stage of a space decoder. If used at this time by the system, note that there is a token of the same number.

[0482]26. The discrete inverse cosine transform (IDCT) based on discrete

inverse-cosine-transform this invention elongates the data related to the frequency of the DC component of the picture concerned. When a specific picture is being compressed, it is quantized and the frequency (pitch) of the light in a picture decreases the entire volume of the information which needs to be memorized. Using this quantized data, IDCT elongates it and returns it to frequency information.
[0483]IDCT acts on the portion of the picture whose size is 8x8 pixels of **. The mathematics applied to this data is mainly managed by the specific standard used in order to code data. However, the important use in this invention is making mathematical functions common between standards, in order to avoid unnecessary duplication of a circuit. By using specific scaling order, the symmetry between the upper part of an algorithm and the lower part is increased, and the reuse of the common mathematics function which eliminates the additional necessity for a circuit by it can be carried out.
[0484]IDCT answers many multiplex standard tokens. The 1st portion of IDCT checks ingress data, in order to guarantee that the size of a data token is right to processing. If an error is not so large, a token stream can actually be corrected by conditions.
[0485]27. The buffer manager of buffer manager this invention receives ingress moving image information, and supplies the information about the timing of data arrival, a display, and a frame rate to an address generator. A multiple buffer is used in order to give an indication and to enable change of both the rates of a display. Generally a display and a display rate change according to the coded data and the monitor by which information is being displayed. A data arrival rate changes according to the error included in the raw material generally used in order to make coding, decryption, or data. Information will be elongated if a buffer manager is reached. However, although data is useful for an expansion circuit, it is arranged for a specific display unit in use by order which is not useful. If the block of data is inputted into a buffer manager, a buffer manager will supply information to an address generator so that the block of data can be arranged in an order which can use a display device. Thereby, frame rate conversion required in order to adjust the data block which carries out ingress so that a data block can be displayed on a specific display device in use is used for a buffer manager.
[0486]In this invention, a buffer manager supplies information mainly to an address generator. However, it is also required to interface other elements of a system. For example, there is an interface with input FIFO which transmits a token to the buffer manager which writes in these tokens as a result and is supplied to an address generator.
[0487]A buffer manager interfaces also with a display address generator, and receives the information about whether the preparation which displays data with a new display device is complete. Similarly, a buffer manager checks what the display address generator cleared the information for the display from a buffer for.

[0488]The buffer manager of this invention memorizes whether a specific buffer is empty, it is filled, use preparation is complete, or it is under use, and memorizes the number of displays relevant to the specific data in each buffer further. Thus, a buffer manager determines the state of a buffer selectively by changing only one buffer into the state where display preparation was completed, at a stretch. When a buffer is displayed, a buffer is once in an "empty" state. A buffer manager will be determined about the state of each buffer, and acceptance preparation of new data, if a picture start, a flash plate, an effective token, or an access token is received. For example, in order to find the buffer which can receive new data, a buffer manager is made to circulate through a picture start token through each buffer.
[0489]A buffer manager can be constituted so that the multiplex standard necessary condition directed by the token which it receives may be treated. For example, in an H.26I. standard standard, data can be skipped during the display. When this kind of token reaches a buffer manager, the flash plate of the data which should be skipped is carried out from the buffer with which it is memorized.
[0490]Thus, data can be effectively displayed by managing a buffer according to the display device of the compression standard used in order to code the data concerned, the speed by which data is decrypted, and a particular type in use.
[0491]An old description for the usual expert in the technical field concerned All the subordinate features, the details of sufficient grade to realize the purpose and this invention provided with the advantage -- with, it is sure that realization of an overall concept and a system and the operation of various modes of this invention are explained appropriately. However, in order to make it easy to understand still more thoroughly the additional details relevant to the much more concrete and commercial realization method of various examples of this invention and this invention, ****** which continues and shows the much more description and explanation below is preferred.
[0492]The following is more detailed explanation of the chip set of a multiplex standard animation decoder. Explanation is divided into three portions, A, B, and C.
[0493]Suppose that added disclosure is again shown in the following sections for systematization of explanation, clear-izing, and convenient-izing.
[0494]- Description about the feature common to the chip of a chip set.
[0495]- Token, description section of description and the time decoder chip of 2 line interface, a DRAM interface microprocessor interface clock, and a space decoder chip
A.1 "Utilizing method of this description"
The 1st description section is aimed at most problems of the electrical design relevant to use of the chip set.
[0496]A.1.1 "Typographical custom"
In order to emphasize the information on a certain kind of class, some typographical custom is used. The example is shown below.

[0497]The name wire name active high signal of a token
Wire name active low signal
Register name section A.2 ″Animation decoder ******-″
* Animation data rate *MPEG resolution to coded data rate *21 MB/s to 30-MHz operation * decoding MPEG and JPEG & H.261*25 Mb/s 704x480, 30 Hz, The pin PQFP package * independent coded data of whole chroma sampling format *JPEG base-line decryption * glue loess page mode DRAMA interface *208 which has * flexibility to 4:2:0, And a decoder clock * reorder MPEG picture sequence animation decoder family provides the low chip-counts solution for realizing a high-resolution degree digital animation decoder. Now, a chip set can be constituted so that three different animations and a picture encoding system, i.e., JPEG, MPEG, and H.261 may be supported.
[0498]All the JPEG base-line picture decryption is supported. 720x480, 30 Hz, and a 4:2:2 JPEG-coding animation can decrypt in real time.
[0499]CIF (common exchange format) and a QCIF H.261 animation can be decrypted. 740x480, 30 Hz, and all the ****** MPEG animations with the format to 4:2:0 can be decrypted.
[0500]Notes: The aforementioned value was only shown as an example and does not necessarily show the limited condition of one example of this invention. Therefore, please understand not interfering, even if it uses other values and/or the range.
[0501]A.2.1 ″System configuration″
A.2.1.1 ″Output-format-izing″
In each example shown below, it is required that the form of an output formatting part should be reformatted to using the data supplied to the output of a space decoder or a time decoder depending on the case and a computer, or a display system. The details of this formatting change with applications. In being easy as an example, it only needs the address generator for writing in the data output in order of a raster using the data output by which Brooke formatting was carried out with a decoder chip.
[0502]An image formatting part is one single chip VLSI device which provides an output format-ized function [the wide range].
[0503]A.2.1.2 ″JPEG still picture decryption″
One single space decoder provided with DRAM which is not an off-chip can carry out decryption possible [of the base-line JPEG image] at high speed. A space decoder supports all the functions of base-line JPEG. However, the image size which can be decrypted is restricted by the size of the output buffer supplied by the user, and may also have things. The characteristic of an output formatting part can restrict a chroma sampling format and the color space which can be supported.
[0504]A.2.1.3 ″JPEG animation decryption″
By adding off-chip DRAM to a space decoder, it is possible to decrypt a JPEG coding animation in real time. It depends on the rate of an animation and coded data for the

size and speed of a buffer which are needed. The time decoder does not need to decrypt a JPEG coding animation. However, when a time decoder is on a multiplex standard decoder chip set and the system is constituted for JPEG operations, data is only supplied without change or correction via a time decoder.
[0505]A.2.1.4 ″H.261 decryption″
In order to realize an H.261 animation decoder, both are required for a space decoder and a time decoder. When operating with a low coding data rate in a small picture format, the DRAM interface on both devices can be constituted so that the quantity of DRAM needed for suitable operation can be reduced. Generally, each space decoder and a time decoder need one single 4Mb(for example, 512kx8) DRAM.
[0506]A.2.1.5 ″MPEG decryption″
The configuration needed for MPEG operation is the same as the case of H.261. However, in order to support an available and also large picture format in MPEG so that he can understand if it is the usual expert in the technical field concerned, a still larger DRAM buffer may be required.
[0507]Section A.3 ″Token″
A.3.1 ″Token format″
When following this invention, a token provides an extensible format, in order to transmit information via a decoder chip set. In this invention, although the minimum width of each word of a token is 8 bits, if it is the usual expert in the technical field concerned, he should understand not interfering, even if the width of a token is arbitrary. One token is [/ 1 or two or more words] extensible, and this is attained using the extension bit in each word.
[0508]It is shown whether a token continues an extension bit in other words. All the words other than the word of the last of a token are set to 1. When the 1st word of a token has an extension bit of 0, this shows that the length of the token concerned is only 1 word.
[0509]Each token is identified by the address field which it starts in the bit 7 which is the 1st word of a token. The length of an address field is variable and potentially extensible (however, since there is no address with which length exceeds 8 bits in the present chip). [/ two or more words] If it is the usual expert in the technical field concerned, he should understand the length of an address being arbitrary and obtaining it also in this case.
[0510]More bits than 8 of data are transmitted depending on an interface. For example, the width of the output of a space decoder is 9 bits (an extension bit is included and it is 10 bits). The only token using these additional bits is a data token. The data token can have a bit required in order to process in the specific place of a system, and a bit of the same number. Other tokens of all the disregard an additional bit.
[0511]A.3.2 ″Data token″
A data token carries data to the following stage from one processing stage. Therefore,

the characteristic of this token changes as it passes a decoder. Change by where [in a system] the meaning of the data carried by the data token has the data token concerned, namely, data is positional dependence. According to where [in a space decoder] the data token concerned carries out the whereabouts of the data about this point, they may be either frequency domain data or pixel area data. For example, in the input of a space decoder, a data token has the bit serial coding video data packed to an 8-bit word. In this point, there is no restriction in the length of each token. however, in the output of a space decoder, the width of each word of each data token is 9 bits by contrast -- it has a word of 64 exactly.
[0512]A.3.3 ″Use of token formatting data″
It is required to carry out direct continuation of the circuit to the input or output of a decoder or a chip set depending on a use. When the most, data tokens are collected and it is enough to detect the token of 2 and 3 which provides synchronization information (it is (for example, like PICTURE START)). Please refer to the following sections A.16 ″connection with the output of a space decoder″, and A.19 ″connection with the output of a time decoder″ about this point.
[0513]As already stated, in order to identify when each new token starts, it is enough just to observe the activity about an extension bit. Again, an extension bit transmits the word of the last of the present token. An address field can be tested in order to identify a token. The token which was unnecessary or has not been recognized is possible (and cancellation is possible) in consumption, without getting to know those contents. However, the recognized token makes suitable action cause.
[0514]The data input to a space decoder can be supplied in a data token as a byte of the coded data (refer to section A.10 ″coded data input″). Supplying a token via a coding dataport or a microprocessor interface makes it possible for many features of a decoder chip set to comprise a data stream. This provides the alternative of carrying out a configuration via a microprocessor interface.
[0515]
[Table 1]
[0516]
[Table 2]
A.3.4 ″Explanation of a token″
In this section, the token realized in a space decoder and a time decoder chip is explained in detail based on this invention.
Meaning [and] the bit by which notes: ″r″ is reserved now, a value is 0. Unless it is mentioned specially, all the integers have no numerals.
[0517]
[Table 3]
[0518]
[Table 4]

[0519]
[Table 5]
[0520]
[Table 6]
[0521]
[Table 7]
[0522]
[Table 8]
[0523]
[Table 9]
[0524]
[Table 10]
[0525]
[Table 11]
A.3.5 ″Number transmitted in a token″
A.3.5.1 ″Ingredient identification number″
The ingredient ID number based on this invention is 2 binary integers which specify a color component. Generally these two bit fields are arranged as a part of header in a data token.In the case of MPEG and H.261, correlation is dramatically easy.
[0526]
[Table 12]
Since the color component which can be used in JPBG is not restricted, in the case of JPEG, conditions are still more complicated. A decoder chip makes possible a color component which is different to a maximum of four sorts in each scan. ID is assigned one by one as the specification of a color component reaches a decoder.
[0527]A.3.5.2 ″Level and vertical sampling number″
About each of four sorts of ****, there is specification about the number of the blocks arranged horizontally and vertically in a macro block. Only one small 2 binary integers are contained in this specification from the block count including the integer of 2 bits. For example, the arrangement of ingredient ID of the 4:2:0 chroma sampling (drawing 45) in MPEG (or H.26 l.) is shown in Table 13.
[0528]
[Table 13]
The ingredient assignment to ingredient ID changes with uses about JPEG and a 4:2:2 chroma sampling. Refer to section A.3.5.1.
[0529]Notes: When 4:2:2 data is being processed, JPEG needs 2:1:1 structures to the macro block.
[0530]
[Table 14]
A.3.6 ″Special token format″

It is based on this invention, for example, a data token and a QUANT table token are used for those ″extended forms″ in a decoder chip set. In extended form, a token contains some data. In the case of a data token, this seed token can contain coded data or pixel data. In the case of a QUANT table token, these tokens contain quantizer expression news.
[0531]″The non-extending type″ of these tokens is defined as ″being empty″ in this invention. This token format provides the place continued and filled by the extended version of the same token in a token stream. This format can be applied mainly to an encoder, therefore does not explain this in detail any more.
[0532]
[Table 15]
[0533]
[Table 16]
A.3.7 ″Use of the token to a different standard″
The subset from which the token defined based on this invention differs is used for each standard.
[0534]Section A.4 ″2 Line interface″
A.4.1 ″2 Line interface and token port″
In a chip set, in order to control an information flow, easy 2 line effective / acceptance protocol are used. When it is observed that clock generation preparation is complete in the transmitting side and a receiver, between blocks, data is only transmitted.
[0535]1) the transmitting side is not ready when the transmitting side in which the 3 transmitting side in which data transfer 2 receiver is not ready is not ready is not ready (above 3) -- it must wait for the input of a receiver to a case. When a receiver is not ready (above 2) and a receiver is not ready, the transmitting side continues supplying the same data as the output until it is accepted by the receiver.
[0536]When token information is transmitted between blocks, 2 line interface during a block is called a token port.
[0537]A.4.2 ″Service space″
In order to connect three chips, 2 line interface is used for the decoder chip set based on this invention. The coded data input to a space decoder is 2 line interface similarly.
[0538]A.4.3 ″Bus signal″
The width of the data word transmitted by 2 line interface changes according to the necessity for a related interface (refer to drawing 44 ″token on an interface larger than 8 bits″). For example, the coefficients of 12 bits are few, or only output 9 bits and are inputted into a discrete inverse-cosine-transform machine (IDCT).
[0539]
[Table 17]
In addition to a data signal, there are other three signals sent via 2 line interface.
[0540]* Valid signal (valid) * acceptance signal (accept) * extension signal (extension)

A.4.3.1 ″Extended signal″
An extended signal corresponds to the already described token extension bit.
[0541]A.4.4 ″Design consideration matter″
2 line interface is an object for communication one by one between short distance and a chip.
[0542]In order to make the length of the PCB track between chips into the minimum, a decoder chip approaches mutually and must be arranged. The length of a track must be held as much as possible regarding the place at 25 mm or less.PCB track capacitance must be held at the minimum.
[0543]Distribution of a clock must be designed make the clock through during a chip into the minimum. When there is a certain clock through, a ″reception chip″ must arrange so that a clock may be seen before a ″transmitting chip.″
[0544]All the chips which communicate via 2 line interface must operate from the same digital power source.
[0545]A.4.5 ″Interface timing″
[0546]
[Table 18]
Notes: Drawing 47 shows 2 line interface between a system demux chip and the coding dataport of the space decoder which operates from a main decoder clock. Since this 2 line interface can operate from the coding data clock which does not interfere to a decoder clock even if asynchronous, this is an option. Please refer to the section A.10.5 and ″the coded data clock.″ Similarly, the display interface of an image formatting part can operate from an asynchronous clock to a main decoder clock.
[0547]A.4.6 ″Signal level″
2 line interface uses a COMS input and an output. VIHmin is about 70% of VDD, and VILmax is about 30% of VDD. The value shown in Table 19 is a value of VIH and VIL when VDD is the worst. VDD=5.0**0.25V.
[0548]
[Table 19]
A.4.7 ″Control clock″
Generally, the clock which controls transmission via 2 line interface is a decoder clock of a chip. The coding dataport input to a space decoder is an exception. This is controlled by an encoding clock. Suppose that it explains further here about a clock signal.
[0549]Section A.5 ″DRAM interface″
A.5.1 ″DRAM interface″
One single high performance and the DRAM interface which can be constituted are used in each animation decoder chip. Generally, the DRAM interface on each chip is substantially the same. However, interfaces differ mutually in the method of treating a

channel right of priority. The interface is designed drive directly DRAM used by each decoder chip. Generally, external logic, the buffer, or the component is unnecessary in order to connect a DRAM interface to DRAM in most systems.
[0550]A.5.2 ″Interface signal″
[0551]
[Table 20]
Based on this invention, an interface can be constituted in two methods shown below.
[0552]* The detailed timing of an interface can be constituted so that DRAM different type [various] may be accommodated.
[0553]* ″Width″ of a DRAM interface can be constituted so that the cost / performance trade-off in different application may be provided.
[0554]A.5.3 ″Composition of a DRAM interface″
Generally, are, namely, three groups of the register relevant to a DRAM interface are an interface timing configuration register, a bus configuration register, and a RIFURESHU configuration register. The register shown in the RIFURESHU configuration register table 23 must be constituted at the end.
[0555]A.5.3.1 ″Conditions after reset″
In after reset, the DRAM interface based on this invention starts operation using 1 set of default timing parameters (it corresponds to the latest operating mode). Introduction and a DRAM interface perform continuously a RIFURESHU cycle (except for other the transmission of all the). This is continued until a value is entered in a refreshment interval. Next, the DRAM interface can transmit other types between RIFURESHU cycles.
[0556]A.5.3.2 ″Birth control figuration″
The register in the bus configuration 22 is restricted when the data transfer by an interface is not tried, and it should be performed. Just after reset, an interface is placed by this condition, before entering a value in a refreshment interval. An interface can be restricted when transmission is not tried, and it can be reconstructed later if needed. Refer to a time decoder chip access register (A. 18.3.1) and a space decoder buffer manager access register (A. 13.1.1).
[0557]A.5.3.3 ″Composition of interface timing″
Based on this invention, correction of interface timing configuration information is controlled by an interface timing access register. if 1 is entered in this register -- the interface timing register 21 -- being shown -- it becomes correctable. While being interface timing access =1, the DRAM interface continues operating by the configuration of the before. After filling in 1, before filling in arbitrary interface timing registers, the user has to wait from interface timing access until ****** of 1 becomes possible.
[0558]Completion of a configuration must enter 0 in interface timing access. Next, a new configuration is transmitted to a DRAM interface.

[0559]A.5.3.4 "Refreshment configuration"
The refreshment interval of the DRAM interface concerning this invention can be constituted only once following reset. An interface performs a refresh cycle continuously until a refreshment interval is constituted. this -- all -- others -- data transfer is prevented. Data transfer can be started after a value is entered in a refreshment interval.
[0560]After DRAM generally switches on a power supply first, "a pause (pause)" is needed between 100 microseconds and 500 microseconds, so that it may be common knowledge in the technical field concerned, and before a normal system operation becomes possible, many refresh cycles follow. Therefore, before entering a value in a refreshment interval, these DRAM start necessary conditions must be satisfied.
[0561]A.5.3.5 "Read access to configuration registers"
All the DRAM interface registers of this invention can be read.
[0562]A.5.4 "Interface timing (tic)"
This clock obtained from the clock which runs DRAM interface timing in the clock rate of a device (decoder clock) 4 times the rate of an input is generated by PLL on chip.
[0563]In order to explain briefly, the cycle of this high-speed clock is called a tic.
[0564]A.5.5 "Interface register"
[0565]
[Table 21]
[0566]
[Table 22]
A.5.6 "Interface operation"
A DRAM interface uses a fast page mode. Access of three different types is supported.
[0567]* Read (lead).
* Write in (light).
* the lead of each refreshment or right access transmits one burst up to 1 to 64 bytes to one single DRAM page address. Each continuous access is treated as random access to a new DRAM page, without mixing a lead and light transmission in one single access.
[0568]
[Table 23]
A.5.7 "Structure of access"
Each access has two portions.
[0569]* In access start * data transfer this invention, each access starts with an access start, and one or more data transfer cycles follow it. The lead of both an access start and a data transfer cycle, a light, and refreshment modification (variant) occur.
[0570]If the data transfer of the last to specific access is completed, an interface will

stop at this state until it goes into that default (A. 5.7.3 references) and start preparation of new access is completed. When the last access is completed and start preparation of new access is completed, new access starts promptly.
[0571]A.5.7.1 ″Access start″
An access start provides the page address for a lead or light transmission, and establishes a certain amount of initial signal conditions. There are three different access starts based on this invention.
[0572]* Start [of the start * writing of reading] * Start of refreshment [0573]
[Table 24]
In each case, the timing of RAS and a line address is controlled by register RAS falling and page start length. OE and the state of DRAM data [31:0] are ****(ed) until RAS descends from the end of pre- data transfer. Three different types of an access start only differ in the way these drive OE and DRAM data [31:0], when RAS descends. Reference drawing 52.
[0574]A.5.7.2 ″Data transfer″
There is a data transfer cycle of a different type in this invention.
[0575]* Second-half write-cycle * of a high-speed page read-cycle * high speed page The start of refresh cycle refreshment can follow only by one single refresh cycle. The start of reading (or writing) can follow by one or more high-speed page reading (or writing) cycles. When starting a read cycle, CAS is driven to a high and a new column address is driven.
[0576]An early write cycle is used. WE is driven to a low when starting the first write-in transmission, and remains in a low to the end of the last write-in transmission. Output data is driven by an address.
[0577]Since CAS in front of a RAS refresh cycle is started by the start of a refresh cycle, an interface signal does not work during the refresh cycle. The purpose of a refresh cycle is to suit the minimum RAS low period needed by DRAM.
[0578]A.5.7.3 ″Default of an interface″
The interface signal in this invention goes into a default in the end of access.
[0579]RAS, GAS, and WE are highs.
[0580]* Data and OE stop at the state in front of them.
[0581]* addr maintains the state where it was stabilized.
[0582]A.5.8 ″Data bus width″
Two bit registers and DRAM-data width make it possible to constitute the width of the data path of a DRAM interface. This makes it possible to make DRAM cost into the minimum, when using it in a small picture format.
[0583]
[Table 25]
A.5.9 ″Line address width″
The number of bits taken from the middle section of a 24-bit internal address in order

to provide a line address is constituted by the row address bit register.
[0584]
[Table 26]
A.5.10 "Address bit"
On-chip 24 bit addresses are generated. It is dependent on the number of bits selected by the width and the object for line addresses of the data bus how this address is used in order to form a sequence and a line address. since it is not allowed to use all the internal address bits depending on a configuration -- "-- it hides and bit" is created.
[0585]Similarly, a line address is extracted from the center portion of an address. Therefore, thereby, the speed [RIFURRESHU / DRAM / speed / automatically] becomes the maximum.
[0586]
[Table 27]
A.5.10.1 "Low priority line address bit"
The bits from the 4 [lowest] to 6 of a line (column) address are used in order to provide the address for fast page mode transmission of a maximum of 64 bytes. It depends on the width of a data bus for the number of address bits demanded in order to control these transmission (A. 5.8 references).
[0587]A.5.10.2 "Line address decryption for accessing more DRAM banks"
When only one single bank of DRAM is used, it depends for the width of the line address used on the type of DRAM used. More memories rather than generally being able to supply by one single DRAM banks in the case of necessity and pickpocket application, It is possible to constitute a larger line address, therefore in order to select one single DRAM banks, it is possible to decrypt some line address bits.
[0588]Notes: A line address is extracted from the center of an internal address. In order to select the bank of DRAM, when some bits of a line address are decrypted, the bank in which all the values which these "bank selection bits" can take are among the DRAMs must be selected. When that is not right, a hole is left behind in an address space.
[0589]A.5.11 "-izing of a DRAM interface which can be operated"
In this invention, in order to make all the output signals on a DRAM interface into high impedance, there are two methods. That is, they are the method of setting a DRAMenable register, and a method by a DRAMenable signal. In order to operate the driver on a DRAM interface, both of these registers and signals must be the logic 1. An interface will become high impedance if either is a low.
[0590]Notes: When a DRAM interface is high impedance, data processing on chip is not made to end. Therefore, when a chip tries access to DRAM, an error occurs, and an interface is high impedance simultaneously.
[0591]or [that the capability to make a DRAM interface into high impedance tests

other devices based on this invention] -- or a space decoder (.) Or when a time decoder is not in use, it has in order to use DRAM controlled by a space decoder (or time decoder). sharing of the memory by other devices is [be / it / under / normal system operation period / setting] possible -- as -- it does not have intention.
[0592]A.5.12 "Refreshment"
Unless it becomes incompetent by filling in a no refreshment register, a DRAM interface, interval **** determined by the register refreshment interval -- RIFURRESHU [/CAS/before /RAS / refresh cycle is used, and / DRAM] automatically. Here, /signal name/shows the inversion signal of a signal.
[0593]The value in a refreshment interval specifies the interval (interval) between the refresh cycles in the cycle of 16 decoder clock cycle. The value in the range 1.255 can be constituted. The value 0 is automatically loaded after reset and it forces that a DRAM interface performs a refresh cycle continuously until an effective refreshment interval is constituted (once it becomes operation possible). It is recommended that a refreshment interval is constituted only only once after each reset.
[0594]If /reset/is expressed, a DRAM interface will become impossible [RIFURRESHU / DRAM] simultaneously. However, these decoder chips should be reset, and before the contents of the DRAM next become corrupt, it must be possible [the reset time needed by the decoder chip must be short enough, as a result] to reconstruct a DRAM interface.
[0595]A.5.13 "Signal strength (strength)"
A user can constitute the drive strength of the output of a DRAM interface using 3 bit-register CAS strength, RAS strength, addr strength, DRAM-data strength, and OEWE strength. MSB of this triplet value chooses either a high speed or a low-speed edge rate. Two bits with lower importance constitute the output for different load capacitance.
[0596]The default intensity after reset is 6, and when load of this is carried out by 24 pF, and in order to drive the signal between GND and VDD, an output which needs about 10 ns is constituted.
[0597]
[Table 28]
When an output is appropriately constituted to the load which an output is driving, the output suits AC electrical property specified even as Table 35 from Table 32. When constituted appropriately, an outline match is carried out at the load, therefore the minimum overshooting generates each output after signal ****.
[0598]A.5.14 "Electric specification"
All the information which this section is shown only shows one example of this invention, is indicated as an example, and does not necessarily have a restrictive meaning.
[0599]

[Table 29]
Table 29 only shows the maximum rating about an example. In order to guarantee reliability of operation only especially about this example, the stress of less than the value shown in this table must be used.
[0600]
[Table 30]
[0601]
[Table 31]
A.5.14.1 ″AC characteristics″
[0602]
[Table 32]
[0603]
[Table 33]
[0604]
[Table 34]
[0605]
[Table 35]
When reading from DRAM, a DRAM interface carries out the sample of the DRAM data [31:0] along with a rise of /CAS / signal.
[0606]
[Table 36]
Section A.6 ″Microprocessor interface (MPI)″
A standard byte width microprocessor interface (MPI) is used for all the chips in an animation decoder chip set. However, if it is the usual expert in the technical field concerned, even if it is a microprocessor interface of other width, the similarly usable thing should be understood. MPI operates synchronous with various decoder chip clocks.
[0607]A.6.1 ″MPI signal″
[0608]
[Table 37]
A.6.2 ″MPI electrical-and-electric-equipment specification″
[0609]
[Table 38]
[0610]
[Table 39]
[0611]
[Table 40]
A.6.2.1 ″AC characteristics″
[0612]
[Table 41]

[0613]
[Table 42]
A.6.3 ″Interruption″
In this invention, an ″event″ is a term used in order to show the conditions on chip which a user may observe. An error can be shown by the event or it may be useful for a user's software.
[0614]There are two single bit registers relevant to each interruption or an ″event.″ These are a condition event register and a condition mask register.
[0615]A.6.3.1 ″Condition event register″
Condition event registers are 1-bit reading / writing register.
The value is set to 1 by the conditions generated in a circuit.
A register is set to 1, even if it is only temporary and conditions are the cases where it has passed away now. Therefore, being set to 1 is guaranteed until a register is reset by a user's software (or the whole chip is reset).
[0616]* A register is set to zero by filling in the value 1.
[0617]* Stop in the state where a register is not changed, by entering zero in a register.
[0618]* A register must be set to zero by user software, before this condition carries out ****** generating.
[0619]* The register will be reset to zero about reset.
[0620]A.6.3.2 ″Condition mask register″
the condition event register in which a condition mask register corresponds is set -- a case -- 1-bit reading / writing register of interrupt request which generates possible -- it is . If 1 is entered in a condition mask when the condition event is already set, interrupt request will be published immediately.
[0621]* The value 1 interrupts possible.
[0622]* A register is cleared by zero when resetting.
[0623]Unless it is expressed separately, a block stops operation, after generating interrupt request, and after either a condition event or a condition mask register is cleared, it resumes operation.
[0624]A.6.3.3 ″Event and mask bit″
Refer to Table 61 and Table 141 which are always arranged as a group in the corresponding bit position in the byte in a memory map who continued for an event bit and a mask bit. This enables interruption service software to use the value read in the mask register as a mask as a mask to the value in an event register, in order to identify which event generated interruption.
[0625]A.6.3.4 ″Chip event and mask″
Each chip has one single global event bit which summarizes the event activity on a chip. A chip event register provides OR of all the events on chip which have 1 in those mask bits.

[0626]1 in a chip mask bit enables generation of interruption by a chip. 0 in a chip mask bit prevents all events on chip from generating interrupt request.
[0627]The entry from 1 to 0 to a chip event does not influence. It restricts, when all the events (operation-ized by entering 1 in one of mask bits) are cleared, and a chip is cleared.
[0628]A.6.3.5 "irq signal"
When both the chip event bit and the chip event mask are set, /irq / signal is expressed.
[0629]/irq / signal is low active open collector outputs which need an off-chip pull-up resistor machine. When it is under activity, pulldown [of /irq / the output] is carried out by the impedance of 100 ohms or less.
[0630]It should be understood that the pull-up resistor machine of about 4 kilohms is suitable in the case of almost all uses.
[0631]A.6.4 "Access to a register"
A.6.4.1 "Stopping circuit which accesses possible"
The register of most in this invention can be restricted when the block with which the register is related is stopped, and it can be corrected. Therefore, generally the group of a register is connected with the access register.
[0632]The value 0 in an access register shows that the group of the register relevant to the access register concerned must not be corrected. If 1 is entered in an access register, it will be required that a block should be stopped. However, the value 0 is held until it does not interfere even if a block does not stop immediately, and the block concerned suspends the access register of the block concerned.
[0633]Therefore, it should wait for user software until 1 is read in an access register (in order to require access after filling in 1). In the state where the access register was set to 0, if a user enters a value in configuration registers, the result is unfixed.
[0634]A.6.4.2 "Register holding an integer"
The least significant bit of the arbitrary bytes in a memory map is a bit connected with signal data [0].
[0635]The register holding a larger integral value than 8 bits covers two or the byte position which continued either four in a memory map, and is divided. A byte's order is a "big endian" as shown in drawing 64. However, however, no assumption is set up about an order that the byte to whom assumption is not performed about order is entered in a multibyte register.
[0636]The unused bit in the memory map except the unused bit in the register holding a signed integer returns 0 on the occasion of reading. In this case, sign extension of the most significant bit in a register is carried out. For example, in order to fulfill a 16-bit memory map position (2 bytes), sign extension of the 12 bit registers with numerals is carried out. The 16-bit memory map position holding numerals-less 12 binary integers returns 0 from the most significant bit.

[0637]A.6.4.3 ″Address position by which the keyhole was carried out″
In this invention, the memory map position with small access frequency is arranged behind a ″keyhole.″ Have two related registers, namely, ″keyholes″ is a keyhole address register and a keyhole data register.
[0638]A keyhole address specifies the position in the extended address space. Writing operation is accessed at the position over a keyhole data register which read or was specified with the keyhole address register.
[0639]A related keyhole address register is ***************ed after accessing a keyhole data register. The random access in the extended address space is possible only by entering a new value in a keyhole address register [as opposed to each access for a new value value].
[0640]Even if one chip based on this invention is provided with many memory maps ″by which the keyhole was carried out″, it does not interfere. There is no interaction between different keyholes.
[0641]A.6.5 ″Special register″
A.6.5.1 ″Intact register″
The register or bit indicated ″Not to use it″ is a position in the memory map which was not used for realization of this device. Generally, the value 0 can be read from these positions. It does not influence, even if it enters 0 in these positions.
[0642]If it is the usual mastery person in the technical field concerned, in order to make this kind of product maintain compatibility with the modification products in the future so that he can understand, it is recommended that a user's software must not be dependent on the value read in the intact position. Similarly, when constituting a device, this kind of position should be avoided or should be set to the value 0.
[0643]A.6.5.2 ″Reserved register″
Similarly, the influence under which the register or bit indicated to be ″reserved″ in this invention was document-ized by the operation mode of the device does not do.
[0644]A.6.5.3 ″Test registers″
The register or bit indicated to be ″test registers″ controls various modes of the testing sex of the device concerned. Therefore, these registers do not need to be used for the standard use of the device concerned, and do not need to be accessed by a standard device configuration and control software.
[0645]Section A.7 ″Clock″
In an animation decoder system, it is identifiable in the clock with which many differ based on this invention. The example of a clock is shown in drawing 65.
[0646]Resynchronization (on chip) of it is carried out to a respectively new clock as data supplies between different clock regimes in an animation decoder chip set. In this invention, the maximum frequency of all input clocks is 30 MHz. However, naturally it should be able to be understood that it is usable similarly even if it is other frequency which contains larger frequency than 30 MHz in usual [in the technical field

concerned] if it is a mastery person. In each chip, the microprocessor interface (MPI) operates in asynchronous to a chip clock. The image formatting part can generate the audio clock of the low frequency in sync with the picture rate of the decrypted animation. Therefore, this clock can be used in order to supply an audio / animation synchronization.
[0647]A.7.1 "Space decoder clock signal"
A space decoder has two different (and it may be asynchronous-like) clocked into.
[0648]
[Table 43]
A.7.2 "Time decoder clock signal"
A time decoder has only one clocked into.
[0649]
[Table 44]
A.7.3 "Electric specification"
[0650]
[Table 45]
[0651]
[Table 46]
A.7.3.1 "CMOS level"
A clock input signal is a CMOS input. VIHmin is about 70% of VDD, and VILmax is about 30% of VDD. The value shown in Table 46 is a value of VIH and VIL on the worst conditions of VDD, respectively. VDD=5.0**0.25VA.7.3.2 "Stability of a clock"
In this invention, the clock used in order to drive a DRAM interface and the interface between chips is obtained from an input clock signal. the timing specification for these interfaces -- input clock timing -- the range of **100ps -- **** -- it is assumed that it is a stable thing.
[0652]Section A.8 "JTAG"
It becomes increasingly difficult to inspect connection between parts, for example by a conventional method like the test in a circuit using "BEDDOOBUNAIRU (bed-of-nails)" technique as the installation density of a circuit board becomes high. The joint test action group (JTAG) was formed as solution of an access problem, and a trial of standardization of methodology. The compilation of work of this group is "the standard test access port and the boundary scanning architecture" which are adopted by IEEE as the standard 1149.1 now. A space decoder and a time decoder suit this standard.
[0653]A standard uses the boundary scan chain which carries out the series connection of each digital signal pin on a device. It is a conformed type (transparent) however, in a normal system operation state, in a test mode, a test circuit makes shift in of a test pattern possible, and enables supply of a boundary scan chain at the pin of a device. The consequential signal which appears in a circuit board in the input to a

JTAG device can be scanned and checked by comparatively easy test equipment. By this method, the connection between parts can be tested like the field of the logic on a circuit board.
[0654]All the JTAG operations are carried out via the test access port (TAP) which has five pins. A trst (test reset) pin resets a JTAG circuit, in order to guarantee that a device does not upgrade in a test mode. A tck (test clock) pin is used in order to carry out the clock of the clock in-series test pattern to a tdi (test data input) pin and to carry out a clock from a tdo (test data output) pin. Finally, the operation mode of a JTAG circuit is set by carrying out the clock of the sequence of an applicable bit to a tms (test mode selection) pin.
[0655]The JTAG standard is extensible in order to provide the additional feature in discretion of a chip maker. There are nine user instructions (command) including three JTAG indispensable commands in a space decoder and a time decoder. Addition instructions (extra instruction) enable operation of the internal device test of a specific grade, and provide the flexibility of an additional external test. For example, all the device outputs can be created so that it may float by an easy JTAG sequence.
[0656]Please refer to an individual JTAG application note for the details of the directions for an available function and a JTAG port.
[0657]A.8.1 ″Connection of the JTAG pin in a non-JTAG system″
[0658]
[Table 47]
A.8.2 ″Conformity level of IEEE 1149.1″
A.8.2.1 ″Rule″
Although it comments on the following provision, all the rules suit.
[0659]
[Table 48]
[0660]
[Table 49]
A.8.2.2 ″Advice conditions″
[0661]
[Table 50]
[0662]
[Table 51]
A.8.2.3 ″Terms of the license″
[0663]
[Table 52]
Section A.9 ″Space decoder″
* 30 MHz, the operation * decoders MPEG and JPEG. And an H.261.* coding data rate * animation data rate to 25 Mb/s to 21 MB/s. * flexible chroma sampling format * all interface [JPEG base-line decryption * glue loess DRAM] * -- single -- coded data

and decoder clock * standard page mode DRAM of which +5V power supply *208 pin PQFP package * maximum-electric-power-consumption 2.5W* independence was done. A use space decoder is a VLSI decoder chip in which the composition for using it in various JPEG, MPEG, an H.261 picture, and animation decryption application is possible.
[0664]In the minimum configuration that does not use off-chip DRAM, space decoders are one single chip and a high-speed JPEG decoder. By adding DRAM, a space decoder is enabled to decrypt a JPEG coding animation. 720x480, 30 Hz, and 4:2:2 ″a JPEG animation″ can decrypt in real time.
[0665]By using a time decoder, a space decoder can be used in order to decrypt H.26 l. and MPEG (it is JPEG to the appearance). 704x480, 30 Hz, and a 4:2:0MPEG animation can be decrypted.
[0666]The above-mentioned value is only an example, and it is what was shown as an example, and earnestly, it does not necessarily have a restrictive intention and comments on it being a typical value for one example based on this invention again. Therefore, if it is the usual mastery person in the technical field concerned, he should understand that it is usable in other values and/or **.
[0667]A.9.1 ″Space decoder signal″
[0668]
[Table 53]
[0669]
[Table 54]
[0670]
[Table 55]
[0671]
[Table 56]
[0672]
[Table 57]
A. 9.1.1 ″″nc″ connectionless pin″
The pin to which nc and a label were attached in Table 57 expresses the pin which is not used now. These pins must be changed into the state where it does not connect.
[0673]A.9.1.2 ″VDD and GND pin″
All the V and GND pins which are equipped must be connected to a power supply applicable, respectively so that I may be understood, if it is the usual mastery person in the technical field concerned.Unless V and no GND pins are used correctly, the right operation of a device cannot be guaranteed.
[0674]A.9.1.3 ″Connection of the test pin for a normal system operation″
Nine pins of a space decoder are reserved as an object for internal tests.
[0675]
[Table 58]

A.9.1.4 ″JTAG pin for a normal system operation″
Refer to section A.8.1.
[0676]A.9.2 ″Space decoder memory map″
[0677]
[Table 59]
[0678]
[Table 60]
[0679]
[Table 61]
[0680]
[Table 62]
[0681]
[Table 63]
[0682]
[Table 64]
[0683]
[Table 65]
[0684]
[Table 66]
[0685]
[Table 67]
[0686]
[Table 68]
[0687]
[Table 69]
[0688]
[Table 70]
[0689]
[Table 71]
[0690]
[Table 72]
[0691]
[Table 73]
[0692]
[Table 74]
[0693]
[Table 75]
[0694]
[Table 76]
[0695]

[Table 77]
[0696]
[Table 78]
[0697]
[Table 79]
[0698]
[Table 80]
[0699]
[Table 81]
Section A.10 ″Coded data input″
In order to process the system based on this invention, it must know which animation standard is being inputted. From now on, this system can receive either the token which is carrying out point **, or raw byte data. These raw byte data can be arranged in a token with a start code detector next.
[0700]Therefore, coded data and a configuration token can be supplied to a space decoder via two routes shown below.
[0701]* Coded data input port * microprocessor interface (MPI)
It depends on application and system environment for selection of the route to be used. For example, when a data rate is low, a decoder chip set is controlled and it is possible to use one single microprocessor for the purpose of both which multiplex a system bit stream. In this case, it is also possible to input coded data via MPI. Instead, when a coding data rate is high, coded data must be supplied via a coding dataport.
[0702]It is appropriate to use the mixture of MPI and a coding dataport input depending on application.
[0703]A.10.1 ″Coding dataport″
[0704]
[Table 82]
In the coding dataport based on this invention, it is usable in the two modes, i.e., token mode, and a byte mode.
[0705]A.10.1.1 ″Token mode″
In this invention, when a byte mode is a low, the coding dataport operates as a token port in a normal method, and accepts a token under control of coding validity and coding acceptance. Please refer to the section A.4 for the details of the electrical operation of this interface.
[0706]Simultaneously with data [7:0], coding extn, and coding validity, the sample of the signal byte mode is carried out in the rising edge of coding clock.
[0707]A.10.1.2 ″Byte mode″
However, if a byte mode is a high, in data [7:0], it will be transmitted under control of one byte of data, 2 line interface control signal coding validity, and coding acceptance. In this case, coding extn is disregarded. Then, a byte sets on chip and is assembled by

the data token until an input mode is changed.
[0708]1) The 1st word of the token supplied in token mode (head)
2) Word of the last of the supplied token (coding extn becomes a low)
3) The 1st byte of the data supplied in the byte mode. A new data token is set on chip and created automatically.
[0709]A.10.2 ″Supply of the data through MPI″
A token can be supplied to a space decoder via MPI by accessing a coded data input register.
[0710]A.10.2.1 ″Entry of the token through MPI″
In order to make efficient data transfer possible, into a memory map, grouping of the coding data register of this invention is carried out to 2 bytes. Eight data bits and coded data [7:0] are arranged at one place, and a control register, coding BIZUI, an enable mpi input, and coding extn are arranged at somewhere else. Refer to Table 62 and Table 63.
[0711]When constituted for a token input via MPI, whenever one value is entered in coded data [7:0], the present value of coding extn can be used for the present token, and it is extended. Before the word of the last of all tokens is written down in coded data [7:0], about setting coding extn to zero, it is responsible for software.
[0712]For example, a data token is started, when one is entered in coding extn and it enters 0x04 in coded data [7:0] again. Next, the start of this new data token is supplied to a space decoder for processing.
[0713]The present token is extended whenever the new value of 8 bits is entered in coded data [7:0]. For example, in order to introduce another token, it restricts, when making the present token finish, and coding extn needs to be accessed again. The word of the last of the present token is shown by the 0 entry to coding extn which follows by entry to coded data [7:0] of the word of the last of the present token.
[0714]
[Table 83]
Coding busy must be inspected in order to judge whether the preparation in which an interface receives more data is complete in advance of coded data [7:0] each time.
[0715]A.10.3 ″Switching between input modes″
It is possible to change a data input mode dynamically, on condition that the suitable prior measure is taken. Generally, transmission of the token through one arbitrary route must be completed before a switching mode.
[0716]
[Table 84]
By the 1st byte supplied in the byte mode, a data token header can be set on chip and can be generated. All the bytes further transmitted in the byte mode are attached to this data token until an input mode will change from now on. Please recollect that a data token can contain many bits as it is required.

[0717]An MPI register bit, coding BIZUI and a signal, and coding acceptance show in which interface the space decoder concerned is going to receive data. It is guaranteed by observing these signals correctly that data is not lost.
[0718]A.10.4 ″Acceptance rate of coded data″
In this invention, an input circuit supplies a token to a start code detector (refer to section A.11). A start code detector analyzes data in in-series as a data token bit. The processing normal rate of a detector is 1 bit per clock cycle (coding clock). Therefore, a decoder decrypts 1 byte of coded data every 8 cycles of coding clock. However, for example, when a non-data token is supplied, an adding processing cycle may be needed like [when a start code appears] in coded data. When this kind of event occurs, a start code detector continues for a short time, and acceptance of the information beyond it is impossible and ****.
[0719]After a start code detector, data is supplied at the 1st logic coding data buffer. When this buffer fills, it becomes impossible for a start code detector to accept the information beyond it.
[0720]Therefore, the coded data beyond it (or other tokens) is received via MPI in a coding dataport. In this case, a start code detector continues being in the state where the information beyond it is unacceptable. This is shown by signal code-ized acceptance and a register coding busy state.
[0721]It is guaranteed for a user by using coding acceptance and/or coding busy that no encoded information is lost. However, if it is the usual mastery person in the technical field concerned, when data acceptance of a space decoder is impossible, a system should buffer the coded data which arrives newly, or should be able to understand things (or arrival of new data is stopped).
[0722]A.10.5 ″Coding data clock″
Based on this invention, the coding dataport in a space decoder, an input circuit, and other functions are controlled by coding clock. To a main decoder clock, even if this clock is asynchronous, it does not interfere. Data transfer synchronizes with the decoder clock which can be set on chip.
[0723]Section A.11 ″Start code detector″
A.11.1 ″Start code″
In the technical field concerned, MPEG and an H.261 coding animation stream include the identifiable bit pattern called a start code so that it may be common knowledge. In JPEG, the same function is provided in marker code. A start / marker code identifies the significant portion of the syntax of a coded data stream. The analysis of a start / marker code conducted by a start code detector is the 1st step in purging of coded data. A start code detector is the 1st block in the space decoder which follows an input circuit.
[0724]A start / marker code pattern is designed it be identifiable, without decrypting the whole bit stream. Therefore, according to this invention, a start / marker code

pattern can be used in order to help error recovery and a decoder start. In coded data structure, a start code detector detects an error, and provides the function for giving the start of a decoder.
[0725]A.11.2 "Start code detector register"
As already inquired, many start code detector registers are always used by a start code detector. Therefore, when the start code detector which processes data is processing data, accessing these registers has low reliability. About guaranteeing a stop of the start code detector before access to the register, it is a user's responsibility.
[0726]Register start code-ized etectoraccess is used in order to stop a start code detector and to enable access to the register by extension. A start code detector stops, after generating interruption.
[0727]About saying when start code search and abandonment of all the data modes can be started, there are constraints further. About these constraints, it is ending with description A.11.8 and A.11.5.1.
[0728]
[Table 85]
[0729]
[Table 86]
[0730]
[Table 87]
[0731]
[Table 88]
[0732]
[Table 89]
[0733]
[Table 90]
A.11.3 "Conversion to a token from a start code"
The function of the start code detector in a normal system operation is identifying the start code in a data stream and changing into the start code token which corresponds these codes to the next. When the easiest, data is supplied to a start code detector as one long single data token. The output of a start code detector is a data token with briefer a large number which sank below the start code token and have been arranged.
[0734]Instead, based on this invention, the input data to a start code detector can be divided into a data token with briefer a large number. When n is made into an integer about the method of dividing coded data into a data token, there are no restrictions besides saying that each data token must contain 8xn bit.
[0735]Direct supply is possible for other tokens to the input of a start code detector. In this case, a token is supplied to other stages of a space decoder via a start code detector, without processing. Into coded data, these tokens can be restricted just

before a start code position, and can be inserted.
[0736]A.11.3.1 ″Start code format″
According to the start code detector of this invention, three different start code formats are recognized. This is constituted via a REJISUTASUTATOKODO detector coding standard.
[0737]
[Table 91]
A.11.3.2 ″Start code token equivalent″
The place and start code detector which detected the start code examine the value relevant to the start code concerned, and generate a suitable token. Generally, a token is connected with related MPEG syntax and named. however, if it is the usual mastery person in the technical field concerned, a token should understand following an additional naming format -- it is. The encoding specification selected now constitutes the relation between a start code value and the generated token. This relation is shown in Table 92.
[0738]
[Table 92]
A.11.3.3 ″Expanded function of encoding specification″
According to encoding specification, encoding specification provides the mechanism of a large number to which the use makes possible data embedding into a data stream unsettled now. Even if this is possible also for regarding it as the specific ″user-datum″ application which provides a specific maker with an additional usage easy, instead regards it as ″extended data″, it does not interfere. The right to use extended data as an encoding specification authority in order to add a function to encoding specification in the future was reserved.
[0739]Two clear mechanisms are used. A marker code is made to precede with the block of a user and extended data in JPEG. However, in H.261, the ″additional information″ shown by the additional information bit in coded data is inserted. The art of these both sides can be used in MPEG.
[0740]If this invention is followed, the MPEG/JPEG block of a user and extended data with which it is preceded by a start / marker code is detectable with a start code detector. ″Additional information″ of H.261/MPEG is detected by the Huffman decoder of this invention. A. Please refer to 14.7 ″reception of additional information.″
[0741]A register, discard extension data, and the discard user datum can constitute a start code detector so that an user datum and extended data may be abandoned. It can access, when this data is not abandoned with a start code detector, and this data reaches an animation demux. A. Please refer to l4.6 and ″reception of a user and extended data.″
[0742]The space decoder of this invention supports the base-line function of JPEG. It is considered that the non-ground line function of JPEG is extended data based on a

space decoder. Therefore, all the JPEG marker codes preceded with data are treated as extended data for non-base-line JPEG.

[0743]A.11.3.4 "Definition of a JPEG table"

JPEG supports Huffman and the quantizer table which were downloaded. The definition of these tables is preceded by the marker codes DNL and DQT in JPEG data. A start code detector generates the tokens DHTMARKER and DQTMARKER, when these marker codes are detected. It is shown that these tokens contain the coded data in which the data token which follows describes Huffman or a quantizer table to an animation demux (the format described in JPEG is used).

[0744]A.11.4 "Detection of an error"

The start code detector can detect the specific error in coded data, and provides some functions which enable recovery by a decoder after an error is detected (A. refer to 11.8 and "search of a start code").

[0745]A.11.4.1 "Count of unjust JPEG length"

Most JPEG marker codes have the count field of the 16-bit length relevant to them. In this field, what quantity shows whether data is connected with this marker code. The length count of 0 and 1 is unjust. Unjust length is restricted and generated when following a data error. In this invention, when the illegal length count mask is set to 1, generating of unjust length generates interruption.

[0746]Since the recovery from the error in JPEG data is difficult to search a start code in JPEG, it is conjectured to need additional application specific data. Refer to paragraph A.11.8.1.

[0747]A.11.4.2 "Duplication of a start / marker code"

In this invention, duplication of a start code is restricted and generated, when following a data error. An MPEG arrangement byte duplication start code is shown in drawing 73. In this case, a start code detector looks at the pattern which appears like a picture start code first. Next, a start code detector sees this picture start code overlap with the group start. Therefore, a start code detector generates a duplication start event. When the overlapping start mask is set to 1, a start code detector interrupts and generates a stop.

[0748]Which of two start codes is unable to judge which cause the right and a data error was. However, the start code detector based on this invention abandons the first start code, and after a duplication start code event is served, it continues decryption of the 2nd start code "that the 2nd start code makes a right thing." When a series of duplication start codes carry out the whereabouts, a start code detector cancels all these [except the last code] codes (the event about each duplication start code is generated).

[0749]In a non-arranging byte system, the same error is possible (H.261, or probably ** MPEG). In this case, the state of an IGUNOA non aryne must be considered similarly. Although the first start code that drawing 74 found is an arrangement byte,

the example which overlaps with a non-arranging start code is shown. When the IGUNOA non aryne is set to 1, the 2nd duplication start code is treated as data by a start code detector. Therefore, a duplication start code event is not generated. Thereby, the data-communications error which may have been generated is hidden. When the IGUNOA non aryne is set to 0, a start code detector looks at the 2nd non-arranging start code. And it is seen overlap with the 1st start code.
[0750]A.11.4.3 ″Non recognition start code″
The start code detector can generate interruption, when the start code which is not recognized is detected (if it is onlay KOGUNAIZUDO start mask =1). Register start value to reading is possible for the start code value leading to this interruption.
[0751]Start code value 0xB4 (sequence error) is used in an MPEG decoder system, in order to show a channel error or a media error. For example, when the error which cannot be corrected is detected, this start code may be inserted in data by the ECC circuit.
[0752]A.11.4.4 ″Event generation order″
A specific coding data pattern (error condition is shown in many cases) makes one or more in the above-mentioned error condition start during a short period in this invention. Therefore, a start code detector shows an investigation order below for coded data about error condition.
[0753]1) When non-arranging start code 2 duplication start code 3 an unrecognized start code therefore, and a non-arranging start code overlap with the start code in the direction of another back, the first generated event is connected with a non-arranging start code. After this event is served, it continues and operation of a start code detector detects a duplication start code behind a short period of time.
[0754]After the test of all the un-arranging and duplication start codes is completed, a start code detector tries to only recognize a start code.
[0755]A.11.5 ″A start and stop of a decoder″
A start code detector makes it possible to complete the present decryption task thoroughly, and provides the function which enables the start of a new task.
[0756]Since the data segment can contain the value which emulates a marker code, using such art using a JPEG coding animation has restriction (A. 11.8.1 references).
[0757]A.11.5.1 ″Clear end of decryption″
If the data about the present picture is completed, a start code detector can be constituted so that interruption and a stop may be generated. This is performed by setting stop after picture =1 and stop after picture mask =1.
[0758]Once the termination of a picture supplies a start code detector, a flash plate token will be generated (A. 11.7.2), and interruption will be generated, and a start code detector will stop. The just completed picture should be careful of being decrypted by the usual method. It may be appropriate to detect the flash plate which reached the output of the decoder chip set depending on application. Arrival of this output shows

the end of the present animation sequence. For example, the display can adhere to the last picture output.
[0759]When a start code detector stops, there may also be data from the ″old″ animation sequence by which the ″trap″ is carried out to the buffer realized by the user between media and a decoding chip. When register discardall data is set, a space decoder is made to make consume it and cancel this data. This is continued until a flash plate token reaches a start code detector or discardall data is reset via a microprocessor interface.
[0760]In this stage, abandonment of all the data from an ″old″ sequence will tidy the preparation in which a decoder starts operation according to a new sequence.
[0761]A.11.5.2 ″Cancellation (discard) start period in all the modes″
All the mode cancellation starts immediately after entering 1 in a discardall data register. A result cannot be predicted, if this is performed when the start code detector is processing data actively.
[0762]After either of the start code detector events (non-arranging start event etc.) generates interruption, all the mode cancellation can be started safely.
[0763]A.11.5.3 ″Start of a new sequence″
When it is strange where [in a certain coded data] a new coding animation sequence starts, a start code search mechanism can be used. This mechanism abandons all the unnecessary data preceded with the start of a sequence. A. Please refer to 11.8.
[0764]A.11.5.4 ″Jump between sequences″
This section shows the application of some aforementioned art. It is what the purpose ″is jumped for″ to other portions in which one coding animation sequence will be involved in part. In this example, a filling system only enables access to a ″block″ of data. This block structure can also be acquired from the sector size of a disk or a blocking error correcting system. Therefore, the position of the entrance in a coding video data and an exit may be unrelated to filing system block structure.
[0765]A stop after picture and a discardall data mechanism enable cancellation of the unnecessary data from an old animation sequence. The discardall data mode is reset by inserting a flash plate token after the end of the last filling system data block. Next, the start code search mode is used in order to abandon all the data in the following data block preceded with a suitable entrance.
[0766]A.11.6 ″A byte's alignment″
In the technical field concerned, a different coding plan has a completely different view about byte alignment of the start / marker code in a data stream so that it may be common knowledge. For example, H.261 looks at communication as a serial bit. Therefore, there is no concept about byte alignment of a start code. IGUNOA non aryne = by setting zero, the start code detector can detect the start code of arbitrary bit alignment. Non aryne start mask = start code non-arranging interruption is oppressed by setting zero.

[0767]However, in contrast with this, JPEG was designed as an object for computer environment to which byte alignment is guaranteed. Therefore, the marker code should be restricted and detected when a byte lines up. When encoding specification is constituted as JPEG, an IGUNOA non aryne register is disregarded and a non-aligning start event is never generated. However, in order to guarantee compatibility with a future product, setting IGUNOA non aryne =1 and non aryne start mask =0 is recommended.
[0768]On the other hand, MPEG was designed fulfill the necessity for both communication (serial bit) and a computer (byte inclination) system. The start code in MPEG data must usually be an alignment byte. However, the standard is designed about the start code serial-bit search be possible (an MPEG bit pattern does not look like a start code, unless it is a start code, no matter what bit alignment it may be). Therefore, an MPEG decoder can be designed permit the loss of the byte alignment in serial data communication.
[0769]When a non-aligning start code is found, it is usually shown that the communication error had already occurred. When an error is ″a bit slip″ in a serial-bit communications system, data including this error is already supplied to the decoder. This error is conjectured to cause other errors in a decoder. However, the new data which reaches a start code detector can be continuously decrypted, after this alignment is lost.
[0770]Interruption can be generated when a non-aligning start code is detected by setting IGUNOA non aryne =0 and non aryne start mask =1. The locus of responsibility is dependent on application. No start codes which follow align (until byte alignment is restored). Therefore, after byte alignment is lost, it may sometimes be appropriate to set non aryne start mask =0.
[0771]
[Table 93]
A.11.7 ″Automatic generation of a token″
In this invention, most tokens outputted by the start code detector reflect the syntax element of various pictures and an animation coding standard directly. In addition to a such ″nature″ token, some useful tokens ″invented″ are generated. The examples of the token which can assert these ownership are a picture end and a coding standard. The syntactic difference of a certain grade between encoding specification is removed, and a token is introduced in order to tidy up under error condition.
[0772]This automatic token generation is performed after in-series analysis of coded data (refer to drawing 70 ″start code detector″). Therefore, a system answers equally the token generated by the start code detector which follows detection of the start code in the token by which direct supply was carried out to the input of the space decoder via the start code detector, and coded data.
[0773]A.11.7.1 ″End display of a picture″

Generally, encoding specification does not transmit the end of a picture clearly. However, the start code detector of this invention generates picture - and a token, when the information which shows that the present picture was completed is detected. The token which makes a picture end generate is shown below. A sequence start, a group start, a picture start, a sequence end, and a flash plate.
[0774]A.11.7.2 "Stop after a picture exit option"
When the register stop after picture is set, after picture - and a token supply a start code detector, it stops. However, a flash plate token "pushes" the end of coded data via a decoder, and in order to reset a system, it is inserted after a picture end. A. Please refer to 11.5.1.
[0775]A.11.7.3 "Introduction of the sequence start for H.261"
Refer to Table 92 without a syntax element equivalent to a sequence start for H.261. When the insertion sequence start register is set, a start code detector, One token is introduced, when it guarantees that there is a sequence start token before the following picture start, namely, a start code detector does not look at a sequence start before a picture start. A sequence start is not introduced when one side already exists.
[0776]Don't use this function with MPEG or JPEG.
[0777]A.11.7.4 "Setting out of the encoding specification over each sequence"
It is always preceded with all the sequence start tokens which leave a start code detector by the coding standard token. The present encoding specification of a start code detector is loaded to this token. This sets encoding specification to each whole decoder chip which carried out new, was and was set to animation sequences.
[0778]A.11.8 "Search of a start code"
The start code detector based on this invention can be used in order to search for the start code specific type in a coded data stream. This enables a decoder to re-start the decryption from the level specified in the syntax of some coded data (setting, after abandoning all the data preceded with it). Application for this is shown below.
[0779]* The start of the decoder after jumping in a strange position to a coding (in for example, case of random access) data file.
[0780]* In order to give recovery after a data error, search for the known point in data.
[0781]For example, Table 94 shows the MPEG start code for which it was searched about the start code search of a different configuration. H.26 l. equivalence, and a JPEG start / marker code can be seen to Table 92.
[0782]
[Table 94]
When the value which is not zero is entered in a start code search, a start code detector starts canceling all the data which carries out ingress until the specified start code is detected. Next, a start code search register is reset to 0, and a normal system operation continues it.

[0783]A start code search is immediately started, after the value which is not is entered in a start code search register. ********** [the result] while the start code detector is processing data actively, when this is performed. Therefore, before starting a start code search, the start code detector must be suspended as [all data] it is not processed. When arbitrary start code detector events (non-aligning start event etc.) have just completed generation of interruption, a start code detector is always on this condition.

[0784]A.11.8.1 ″Restriction in the case of using a start code search with JPEG″

Almost all the JPEG marker code has 16 bit length count field relevant to them. This field shows the length of the data segment connected with the marker code. It does not interfere, even if this segment contains the value which emulates a marker code. In normal operation, a start code detector does not look for the start code in these segments of data.

[0785]When the random access to a certain JPEG coded data carries out ″a land being carried out″ to this seed segment, a start code search mechanism cannot be used by high-reliability. Generally, a JPEG coding animation needs additional external information, in order to identify the entrance for random access.

[0786]Section A.12 ″Start control of a decoder″

A.12.1 ″General view of a decoder start″

In a decoder, after coded data becomes available first, only short time is in an animation display and it is usually displayed. Coded data is accumulated on the buffer in a decoder during this delay. By ******(ing) a buffer in this way, it guarantees that a buffer never becomes empty during the decryption, and a decoder guarantees by extension that a new picture can be decrypted by a regular interval.

[0787]Generally, two functions are needed in order to start a decoder correctly. The mechanism which measures the data of the quantity of which 1st was supplied to the decoder is indispensable. There must be a mechanism for preventing the display of a new animation stream in the 2nd. The space decoder of this invention is provided with an output gate near the output, in order to prevent the new animation stream currently outputted from having a bit counter and starting it near the input, in order to measure the data of the quantity of which arrived.

[0788]The complexity for controlling these functions has three levels.

[0789]* After coded data begins to reach a decoder by holding in the state where the * fundamental control * upper control-output gate always opened was always opened, a picture output starts an output gate at an early stage as much as possible. This is suitable, when decrypting a still picture, or when some displays are being delayed according to other mechanisms.

[0790]The difference between fundamental control and upper control is how many short animation streams to be able to accommodate in the buffer of a decoder at the arbitrary times. For most applications, fundamental control is enough. However, upper

control enables user software to give management by the decoder of a start of some very short animation streams.
[0791]A.12.2 ″MPEG animation buffer verifier″
MPEG describes the object for fixed data rate systems ″animation buffer verifier (VBV).″ Before a decoder starts a display for a picture by using VBV information, it becomes possible to ****** the buffer. If it explains again, it will be guaranteed by this ****** that the buffer of a decoder never becomes during the decryption in the sky.
[0792]If it summarizes, each MPEG picture has a vbvdelay parameter. This parameter specifies time required since the coding data buffer of an ″ideal decoder″ fills with coded data, before decrypting the 1st picture. If cautious of the start delay for the 1st picture, the necessary condition of all the pictures which follow is suited automatically.
[0793]Therefore, MPEG specifies the necessary condition about a start as delay. However, in a fixed bit rate system, this delay is easily convertible for a bit count. This is the foundation of the start control action of the space decoder of this invention.
[0794]A.12.3 ″Definition of a stream″
In this application, the term of a stream is used in order to avoid confusion with the term of the sequence in MPEG. Therefore, a stream means the quantity of the video data which is interested for application. Therefore, one stream is many MPEG sequences, or it may be one picture.
[0795]The decoder start function described by this chapter is related to suiting the VBV necessary condition about the 1st picture in one stream. The necessary condition of the succession picture in the stream concerned is suited automatically.
[0796]A.12.4 ″Start control register″
[0797]
[Table 95]
[0798]
[Table 96]
A.12.5 ″Normally open output gate″
An output gate can be constituted so that the state where it opened may be maintained. When a still picture is being decrypted, or when [in order to manage the start of an animation decoder,] other mechanisms of a certain are available, this configuration composition is appropriate.
[0799]The configuration needed after reset is shown below (access to start control logic is obtained by entering 1 in start-up access).
[0800]* * enabling stream which sets off-chip cue =1 = the decoder start event mask register of all * that sets one is set to 0, and guarantee ****** which makes interruption of those registers incompetent (this is a default after reset).
(A.12.7.1 should be referred to by this about the explanation about the reason held in the state where the output gate was opened).
[0801]A.12.6 ″Basic motion″

In this invention, fundamental control of start logic is enough for almost all MPEG animation application. In this mode, the direct communication of the bit counter is carried out to an output gate. When the end of an animation stream supplies an output gate so that it may be directed by the flash plate token, an output gate is closed automatically. The gate concerned maintains the state where it closed until one enabling is supplied by the bit counter circuit, when a stream attains the start bit count.
[0802]The configuration needed after reset is shown below (access to start control logic is obtained by entering 1 in start-up access).
[0803]* coded data -- receiving the prediction range mostly -- bit count PURISU -- set a kale.
[0804]* In order to make this error condition detectable, set counter flushed tooearly mask =1.
[0805]Two interrupt service routine shown below is needed.
[0806]* Animation demultiplexing service for acquiring the vbv delay value for the 1st picture in each new stream.
[0807]* Counter flushed too early service for reacting to this condition.
[0808]When decrypting the new vbv delay for animation streams, this animation demultiplexer (alias animation parser) can generate interruption (namely, when the 1st picture arrives after a flash plate at an animation demultiplexer). The interrupt service routine must calculate the suitable value for bit count targets, and must fill it in. When a bit counter reaches this target, this bit counter inserts one enabling in the short cue between a bit counter and an output gate. If an output gate opens, this gate will remove one enabling from this cue.
[0809]A.12.6.1 ″Start of the new stream immediately after another end of a stream″ Suppose that it is trying to finish, an MPEG stream is called A as an example, and the MPEG stream which it is going to start is called B. A flash plate token must be inserted after the end of A. This token pushes the data of the last of that coded data through decoding, and warns of a new stream being predicted by various sections of a decoder.
[0810]Usually, a bit counter is ending with reset at zero, and A is in the state which already suited the start condition. After a flash plate, a bit counter begins to count the bit in the stream B. If an animation demux finishes decrypting vbv delay from the 1st picture in the stream B, interruption will be generated and composition of a bit counter will be enabled.
[0811]A gate closes as the flash plate which marks the end of the stream A passes through an output gate. A gate maintains the state where it closed until B suits the start condition. For example, when an output gate closes according to the factor of a large number like the start delay for the depth of the stream B and a buffer, B is able to be already ending with conformity at the start condition condition. In this case, one

enabling is standing by in cue, and an output gate is opened immediately. When that is not right, it must wait for it until the stream B suits the start necessary condition.
[0812]A.12.6.2 ″Continuation of a short stream″
The capacity of the cue located between a bit counter and an output gate enables conformity of three individual animation streams on those start conditions, and it is enough to make it possible to wait for a front stream, in order to terminate decryption. In this invention, when a very short stream is being decrypted, it restricts, when very large as compared with the picture format which an off-chip buffer is decrypting, and this condition occurs.
[0813]In drawing 78, the stream A is being decrypted and is opening the output gate. The streams B and C suit those start conditions, and are thoroughly contained in the buffer managed by the space decoder. The stream D is still reaching the input of a space decoder.
[0814]Enabling the stream B and for C is in cue. Therefore, B which the stream A completes can be started immediately. Similarly, C can follow immediately after B.
[0815]When D suits the start target and A is still passing through an output gate, one enabling is added to cue and fills cue. when the end of D passes a bit counter and no INAIBURU is removed [by] from cue (that is, A is passing through an output gate -- carrying out), a new stream should pass a bit counter -- it cannot start. Coded data holds up its hands in an input until A is completed and one enabling is removed from cue, since an output gate is in the state where it opened so that B can supply an output gate.
[0816]A.12.7 ″Upper operation″
Based on this invention, upper control of start logic enables user software to extend without restriction of the length of the-izing cue which can be operated as A.12.6 ″basic operation″ is described. It restricts, when an animation decoder must accommodate a series of short long animation streams rather than A.12.6.2 ″continuation of a short stream″ is described, and control of this level is needed.
[0817]In addition to the configuration composition needed for basic operation of a system, the configuration shown below is needed after reset (access to start control logic is obtained by entering 1 in start up access).
[0818]* Off-chip cue = set one.
[0819]* When one enabling is removed from cue, in order to make interruption operation possible, set acceptance enabling mask =1.
[0820]* When the bit count target of a stream is attained, in order to make interruption operation possible, set target met mask =1.
[0821]Two additional interrupt service routine shown below is needed.
[0822]* when generated by the acceptable-izing interruption * target achievement interruption target achievement interruption target, if there is no service routine of one enabling **** in the-izing cue which can be off-chip operated, it will not become.

[0823]A.12.7.1 ″Operation mode of output gate logic″
When 1 is entered in an enabling stream register, one enabling is loaded to short cue.
[0824]A gate will be closed if a flash plate (the end of a stream is marked) supplies an output gate. When the end of cue has one available enabling, a gate opens and an acceptance enabling event is generated. When an acceptance enabling mask is set to 1, interruption can be generated and one enabling is removed from the end of cue (a register enabling stream is reset).
[0825]However, when an acceptance enabling mask is set to zero, interruption is not generated following an acceptance enabling event, and the enabling concerned is not removed from the end of cue. This mechanism can be used in order to maintain at the state where the output gate was opened as stated to A.12.5.
[0826]A.12.8 ″Bit count″
A bit counter starts a count, after a flash plate token passes this counter. This flash plate token shows the end of the present animation stream. About this point, a bit counter continues a count until it suits the bit count target set in the bit count target register. Next, a target conformity event is generated, a bit counter resets to zero, and it waits for the following flash plate token.
[0827]If a bit counter reaches a maximum count (255), it will stop increment.
[0828]A.12.9 ″-- bit count PURISU -- a kale -- ″
In this invention, the ** (bit count prescale+1) x512 bit of 2 is required to ************* a bit counter once. bit count PURISU -- kales are three bit registers which can hold a value between 0 and 7.
[0829]
[Table 97]
Since tokenization of some elements of an animation stream has already been carried out (for example, start code), the bit count is approximate, therefore contains a non-data token.
[0830]A.12.10 ″Too early counter Flach″
When a flash plate token reaches a bit counter before achievement of a bit count target, the event which can cause interruption is generated (when it is counter flushed tooearly mask =1). If interruption is generated, a bit counter circuit will stop and the data input beyond it will be prevented. After this event occurs, it is the responsibility for user software to determine when an output gate is opened. An output gate can be created so that it may open by filling in 0 as a bit count target. It should restrict, when trying decryption of the animation stream which continues only the picture of 2 and 3, and these situations should occur.
[0831]Section A.13 ″Buffer management″
A space decoder manages two logical data buffers (CDB), i.e., a coding data buffer, and token buffers (TB).
[0832]CDB buffers coded data between a start code detector and the input of the

Huffman decoder. This provides a buffer action to a low data rate coding video data. TB buffers the data between the output of the Huffman decoder, and the input of a space animation decoding circuit (a reverse modeler, a quantizer, and DCT). This 2nd logical buffer enables processing time to include extension, in order to accommodate the processing with the data volume to change of a picture.
[0833]Both buffers are held physically at one single off-chip DRAM array. These addresses for buffers are generated by the buffer manager.
[0834]A.13.1 "Buffer manager register"
It has intention of a space decoder-buffers manager so that it may be only once constituted immediately after resetting a device.During a normal system operation, the necessary condition about a buffer manager's reconstruction is not.
[0835]After reset is removed from a space decoder, it is stopped (a buffer manager access register is set to 1), and a buffer manager waits for configuration-ization. After a register is constituted, buffer manager access can be set to 0 and can start decryption.
[0836]In almost all cases, the register currently used for the buffer manager during the buffer manager operation cannot be accessed by high-reliability. When it is going to access all buffer manager registers, buffer manager access must be set to 1 before it. This makes it indispensable to follow a waiting protocol until the value 1 is read in buffer manager access. In order to supervise the conditions of a buffer, when polling a register like cdbfull and cdbempty, for example, it must take into consideration about the time which reservation and abandonment of access take.
[0837]
[Table 98]
A.13.1.1 "Value of a buffer manager's pointer"
Generally, between a space decoder and off-chip DRAM, data is transmitted as 64 bytes of a burst (the fast page mode of DRAM is used). All the buffer pointers and length registers mean the block of these 64-byte (512 bits) data. Therefore, 18 bit registers of a buffer manager describe a 256 k block linear address space (namely, 128Mb).
[0838]64-byte transmission is independently required from the width (8, 16, or 32 bits) of a DRAM interface.
[0839]A.13.2 "Use of a buffer manager register"
A space decoder-buffers manager has 2 sets of registers which define two similar buffers. A buffer limit register defines the physical maximum community of a memory space. All the addresses are calculated considering this number as law.
[0840]The range of each buffer is defined by two registers (cdbbase and tbbase), i.e., a buffer base, and the buffer length (cdblength and tblength) in the limit of an available memory. In order for a buffer to become usable, all the registers described so far must be constituted before it.

[0841]setting the present state of each buffer to four registers -- **** -- things are made. A buffer read register (cdbread and tbread) shows the offset from the buffer base where data is read next from there. A buffer number register (cdbnumber and tbnumber) shows the quantity of the data held by the buffer now. It is shown whether status bit cdbfull, tbfull, cdbempty, and tbempty are full of a buffer or it is empty.
[0842]A. The unit about all the above-mentioned registers is a 512-bit data block as stated to 13.1.1. Therefore, in order to ask for the number of bits in a coding data buffer, it must multiply by **** 512 read from cdbnumber.
[0843]A.13.3 "Zero buffer"
The application for still pictures (JPEG is used) with which a "real-time" necessary condition is not applied does not need the large off-chip buffer supported by the buffer manager. In this case, in order to supply 128 bit-stream FIFO on chip to a coding data buffer and token buffers, a DRAM interface can be constituted so that a buffer manager may be disregarded (1 is entered in a register zero buffer).
[0844]The buffer option of zero may be suitable for the application which operates operation by the small picture format in a low data rate.
[0845]Notes: Since it is a part of DRAM interface, a register zero buffer must be set [be / it / under / period / of the formation of post reset composition of a DRAM interface / restricting].
[0846]A.13.4 "Buffer operation"
The data transfer performed via a buffer is controlled by a handshake protocol. Therefore, a buffer is full, or when it is empty, it is guaranteed that a data error does not occur. When a buffer is full, the circuit which tries to send data to a buffer is stopped until a space is made to a buffer. When a buffer continues a full state, the stage which it continues processing furthermore it has been arranged upstream of the buffer concerned stops until it becomes impossible for a space decoder to receive data in the input port. Similarly, when a buffer is empty, the circuit which is going to remove data from a buffer stops until data becomes available.
[0847]A. As shown in 13.2, the position and size of coded data and token buffers are specified with a buffer base and a length register. A user is to blame for guaranteeing constituting these registers and that there is no inconsistency in the memory direction for use between two buffers.
[0848]Section A.14 "Animation demultiplexer"
The animation demultiplexer called an animation parser completes the task of changing coded data into the token started with the start code detector. There are four main processing blocks in an animation demultiplexer, namely, they are a parser state machine, the Huffman decoder (ITOD is included), a macro block counter, and ALU.
[0849]A parser or a state machine directs other units according to the syntax of a coding video data. The Huffman decoder changes variable length code-ized (VLO) data

into an integer. A macro block counter holds the track of the picture section decrypted now. ALU performs required arithmetical computation.
[0850]A.14.1 ″Animation demultiplexer register″
[0851]
[Table 99]
[0852]
[Table 100]
[0853]
[Table 101]
[0854]
[Table 102]
[0855]
[Table 103]
A.14.1.1 ″Loading of a register and generation of a token″
Many registers in an animation demultiplexer hold the value related to the parameter usually transmitted in the coded picture/video data directly. For example, a horizontal picture element register corresponds to MPEG sequence header information, horizontal size and a JPEG frame header parameter, and X. These registers are loaded by the animation demultiplexer when the coded data concerned is decrypted. Similarly, these registers are also connected with a token.
[0856]For example, it is connected with a horizontal picture element register, a token, and horizontal size. A token is generated by the animation demultiplexer when coded data is decrypted (or also in case of immediately after). Similarly, a token can be directly supplied to the input of a space decoder. In this case, the value which a token has constitutes the animation demultiplexer relevant to the token concerned.
[0857]
[Table 104]
[0858]
[Table 105]
[0859]
[Table 106]
[0860]
[Table 107]
[0861]
[Table 108]
[0862]
[Table 109]
[0863]
[Table 110]
[0864]

[Table 111]
[0865]
[Table 112]
A.14.2 ″Structure of a picture″
The dimension of the picture in this invention is described as two different units, i.e., a pixel, and a macro block to a space decoder. In both sides, JPEG and MPEG transmit a picture dimension as a pixel. The field of the buffer containing effective data is determined by transmitting a dimension as a pixel. Even if the field concerned in this case is smaller than total buffer size, it does not interfere. The size of the buffer needed by the decoder is determined by transmitting a dimension as a macro block. A macro block dimension must be searched for by the user from the dimension of a pixel. The space decoder register connected with this information is shown below, namely, they are a horizontal picture element, a vertical pixel, a level macro block, and a vertical macro block.
[0866]A space decoder register, the block count h n, v n, max h, max v, and max component id specify composition (coding minimum unit in JPEG) of a macro block. Each is one 2-bit register which can hold a value to within the limits from 0 to 3. All the registers except max component id specify the block counts from 1 to 4. For example, when the register max h holds one, the width of a macro block is 2 blocks. Similarly, max component id specifies the number of different related color components.
[0867]
[Table 113]
A.14.3 ″Huffman table″
A.14.3.1 ″JPEG style Huffman table description″
In this invention, in order to transmit front description between an encoder and a decoder, a space decoder is provided with the Huffman table description via the format used by JPEO. There are two elements, i.e., BITS, and HUFFVAL in each front description. A user is guided at JPEG specification about the perfect description about the method by which a table is coded.
[0868]A.14.3.1.1 ″BITS″
BITS is a table showing the value which describes whether the sign with which it differs [how many] using each length of VLC is coded. Each entry is one 8 bit values. JPEG permits even a maximum of 16 bits as length of VLC. Therefore, 16 entries are shown in each table.
[0869]** -- BITS [0] describes [from which it differs / how many] whether 1 bit of VLC(s) exist, and another side and BITS [1] describe [from which it differs / how many] whether 2 bits of VLC(s) exist.
[0870]A.14.3.1.2 ″HUFFVAL″
HUFFVAL is a table of 8 bit-data value which aligned in order of increase of the length

of VLC. It depends for the size of this table on the number of different signs which can be coded by VLC.
[0871]A Huffman code-ized table describes JPEG specification still in detail about the method coded or decrypted to this format.
[0872]A.14.3.1.3 "Configuration by a token"
A DHT marker precedes with description of the Huffman table used in order to code AC and a DC coefficient in a JPEG bit stream. When a start code detector recognizes a DHT marker, a detector generates a DHT marker token and arranges the Huffman table description in the data token which follows the next (A. 11.3.4 references).
[0873]AC in a space decoder and the composition of a DC coefficient Huffman table can be attained by supplying data and a DHT marker token to the input of a space decoder, and a space decoder is simultaneously constituted to JPEG operation. Since this mechanism constitutes the DC coefficient Huffman table needed for MPEG operation, it can be used, but the encoding specification of a space decoder must be set to JPEG at the same time a table downloads.
[0874]
[Table 114]
A.14.1.4 "Configuration by MPI"
AC and the DC coefficient Huffman table can fill in a register directly via MPI similarly. Please refer to Table 104.
[0875]* The register dc bits0 [15:0] and dc bits1 [l5:0] hold Table 0x00 and the BITS value for 0x01.
[0876]* The register ac bits0 [15:0] and ac bits1 [15:0] hold Table 0x10 and the BITS value for 0x11.
[0877]* The register dc huffval0 [11:0] and dc huffval1 [11:0] hold Table 0x00 and the HUFFVAL value for 0x01.
[0878]* The register ac huffval0 [161:0] and ac huffval1 [161:0] hold Table 0x10 and the HUFFVAL value for 0x11.
[0879]A.14.4 "Different composition for standards"
An animation demultiplexer supports MPEG, JPEG, and the necessary condition of H.261. Encoding specification is automatically constituted by the CODINCSTANDARD token generated by the start code detector.
[0880]A.14.4.1 "H.261 Huffman table"
Since all the Huffman tables needed in order to decrypt H.261 are held at ROM in a space decoder and held in the parser state machine of an animation demultiplexer more particularly, they do not need a user's intervention.
[0881]A.14.4.2 "Structure of an H.261 picture"
H.261 is defined as what supports only two picture formats, i.e., CIF, and QCIF. A picture format in use is transmitted in the PTYPE section of a bit stream. A space decoder's decryption of this data will arrange it in H261 picture-type register and a

picture type token. All the pictures and macro block composition registers are constituted automatically.
[0882]The information in various registers guarantees that it is arranged in those related tokens and other decoder chips (for example, time decoder) are surely constituted by Table 109 - refer to Table 112, and this.
[0883]A.14.4.3 "MPEG Huffman table"
Since the large majority of a Huffman code-ized table required in order to decrypt MPEG is held in ROM in a space decoder (it is said again that it is in a parser state machine), he does not need a user's intervention. A table required in order to decrypt the DC coefficient of a yne TORARU macro block is an exception. one -- chroma -- two tables are needed as an object for RUMA business and other one. These tables must be constituted by user software before decryption starts.
[0884]
[Table 115]
Table 117 shows the sequence of a token needed since the DC coefficient Huffman table in a space decoder is constituted. Instead, the same result is obtained by writing down this information in a register via MPI.
[0885]It is controlled which DC coefficient Huffman table the register dc huffn uses with each color component. Table 116 shows how it must be constituted as an object for MPEG operations. This can be directly carried out by using a MPEGDCH table token via MPI.
[0886]
[Table 116]
[0887]
[Table 117]
[0888]
[Table 118]
[0889]
[Table 119]
[0890]
[Table 120]
A.14.4.4 "Structure of an MPEG picture"
The composition of the macro block defined for MPEG is the same as the structure where it is used by H.261. The dimension of a picture is coded by coded data.
[0891]When it receives standard 4:2:0 operation, the characteristic of a macro block must be constituted as directed to Table 115. Refer to Tables 109 - 112 which can be carried out by filling in a register as having been directed or supplying an equivalent token to the input of a space decoder for this.
[0892]It depends on application for the method used since a picture dimension is constituted. The picture composition register shown in Table 115 before a decryption

start when a picture format is known can be initialized using a suitable value. Instead, a picture dimension can be decrypted from coded data, and since a space decoder is constituted, it can be used. In this case, the user has to support a parser error MPEG sequence. A. Refer to 14.8 ″change in an MPEG sequence layer.″
[0893]A.14.4.5 ″JPEG″
The encoder option of a large number which change substantially the complexity of the control software required in order to operate a decoder is in base-line JPEG. Generally, the space decoder is designed so that the support needed when it suits the conditions shown below may be the minimum.
* The number of the color components per frame is less than five (Nf<=4).
[0894]A.14.4.6 ″JPEG Huffman table″
JPEG enables a Huffman code-ized table to download. These tables are used when decrypting VLC which describes a coefficient. In order to decrypt a DC coefficient, two tables per scan are permitted, and two tables are permitted to AC coefficients.
[0895]They are the shortening format for image data which there was a JPEG file of three different types, namely, was exchange-formatted and was compressed, and the shortening format for tabular data. Both of a definition of all the tables (Huffman, quantization, etc.) required in order to decrypt the image data and the image data which were compressed are contained in an exchange format file. The shortened image data format file omits a front definition. The shortened front format file includes only the definition of a table.
[0896]A space decoder accepts all three formats. However, when all the tables needed are ending with a definition, only the shortened image data file can be decrypted. This definition is feasible via either of the JPEG files of other two types, or it is possible to instead set up a table with user software.
[0897]When using 1 set of Huffman tables where each scans differ, the definition of a table is arranged in coded data before each scan (encoder). These definitions are automatically loaded by the space decoder, in order to use it during the scan period concerned which scans and follows.
[0898]In order to improve the performance of the Huffman decryption, the case of the sign used [specific] is carried out specially. These signs are the runs of the end of the DC coefficient of the size 0, and a block AC coefficient, and the zero AC coefficient of 16. The value for these special cases must be entered in an applicable register.
[0899]A.14.4.6.1 ″Selection of a table″
It is controlled which AC and a DC coefficient Huffman table the registers dc huff n and ac huff n use with which color component. These relations are defined by TDj and the Taj field of scanning header syntax during the JPEG operation.
[0900]A.14.4.7 ″Structure of a JPEG picture″
Two clear levels are among base-line JPEG decryption supported by the space

decoder, namely, it is a larger (Nf>4) level to four ingredients per frame than the level of (Nf<=4), and four ingredients per frame. In using Nf>4, the control software needed is compounded further.
[0901]A.14.4.7.1 "Nf<=4"
Refer to Table 115 which constitutes a macro block composition register on the occasion of these decryption for the frame ingredient specification parameter contained in a JPEG frame header. Since all the specifications required in order to decrypt four different color components are defined, a user's intervention is not needed.
[0902]Please study JPEG specification about the details of the option provided by JPEG. Similarly, the JPEG picture format is described briefly [the section A.16.1].
[0903]A.14.4.7.2 "JPEG with more ingredients than 4"
Decryption of the JPEG file containing the color component (maximum number permitted by JPEG) from which a maximum of 256 differs is possible for a space decoder. However, when it is going to decrypt more color components than four ingredients, additional user intervention is needed. In JPEG, even four ingredients of maximums are only permitted in all scans.
[0904]A.14.4.8 "Non-standard variant"
As already stated, a space decoder supports more picture formats than the format defined by JPEG and MPEG.
[0905]JPEG restricts the minimum coding unit so that the block with more coding units than 10 blocks per one scan may not be included. Since the space decoder can process the arbitrary minimum coding units which can be described by block count hn, the block count v n, max h, and max v, this limit is not applied to a space decoder.
[0906]Refer to Table 115 defined only about 4:2:0 macro blocks in MPEG. However, the space decoder can process other three ingredient macro block structures (for example, 4:2:2).
[0907]A.14.5 "Animation event and error"
The animation demultiplexer can generate the event, i.e., the parser event, and the Huffman event of two types. Please refer to A.6.3 "interruption" about description of an event and how to interrupt and treat **.
[0908]A.14.5.1 "Huffman event"
The Huffman event is generated by the Huffman decoder. The event directed in the Huffman event and the Huffman mask determines whether interruption is generated or not. When the Huffman mask is set to 1, interruption is generated and the Huffman decoder stops. A register Huffman error code [2:0] holds the value which shows the cause of an event.
[0909]When 1 is entered in the Huffman event after serving interruption, the Huffman decoder tries recovery from an error. Similarly, when the Huffman mask is set to 0, (the mask of the interruption being carried out and the Huffman decoder's not being

stopped) and the Huffman decoder try recovery from an error automatically.
[0910]A.14.5.2 ″Parser event″
A parser event is generated by the parser. This event is directed in a parser event. It is determined by the parser mask in it or subsequent ones whether interruption is generated or not. When the parser mask is set to 1, interruption is generated and a parser stops. A register parser error code [7:0] holds the value which directs the cause of an event.
[0911]When 1 is entered in the Huffman event after serving interruption, the Huffman decoder tries recovery from an error. Similarly, when the Huffman mask is set to 0, (the mask of the interruption being carried out and the Huffman decoder's not being stopped) and the Huffman decoder try recovery from an error automatically.
[0912]In entering 1 in a parser event after serving interruption, a parser resumes operation. When an event points to a bit stream error, an animation demultiplexer tries recovery from an error.
[0913]When a parser mask is set to 0, a parser sets the event bit, but interruption or a stop is not generated. A parser continues operation and tries automatic recovery from an error.
[0914]
[Table 121]
[0915]
[Table 122]
[0916]
[Table 123]
[0917]
[Table 124]
[0918]
[Table 125]
[0919]
[Table 126]
The subset from which the defined parser error code differs is used for each standard.
[0920]
[Table 127]
[0921]
[Table 128]
A.14.6 ″Reception of an user datum and extended data″
The same mechanism for embedding an user datum and extended data is used for MPEG and JPEG. Data is preceded by a start / marker code. When application does not get interested in the data concerned, a start code detector can constitute so that this data may be deleted (A. 11.3.3 references).
[0922]A.14.6.1 ″Discernment of a data source″

A parser event, an ERREXTENSION token, and an ERR user token show arrival of the EXTENSION data in an animation demultiplexer, or an user-datum token. When these tokens are generated by the start code detector, refer to Table 92 which has a value of the start / marker code who made the token generate in a start code detector for (A.11.3.3 references) and these tokens. This value can be read by reading a rom revision register, while supporting parser interruption. An animation demultiplexer maintains a halt condition until 1 is entered in a parser event (A. refer to 6.3 and "interruption").

[0923]A.14.6.2 "Reading of data"

EXTENSION data and an user-datum token are predicted to follow promptly by a data token with extended data or an user datum. An animation demultiplexer generates either ERREXTENSION data or an ERR user-datum parser event by arrival of this data token. The 1st byte of a data token can be read by reading a rom revision register, while having served interruption.

[0924]The continuation state of an animation demultiplexer register determines the action mode after the event was cleared. When this register holds the value 0, all the remaining data in a data token is consumed by the animation demultiplexer, and no event is generated. When continuation is set to 1, an event is generated as each expansive byte or user datum reaches an animation demultiplexer. This is continued until a data token is used up or continuation is set to 0.

[0925]Notes: The 1st byte of 1 extension / user datum is always shown via a romrevision register regardless of the state of continuation.

[0926]2) There is no event which shows that read the byte of the last of extension/user datum was finished.

[0927]A.14.7 "Reception of additional information"

H.261 and MPEG make it possible to embed the information which extends encoding specification in the picture of a block (H.261) or a slice (MPEG), and a group. This mechanism differs from the mechanism for extended data and user data (finishing [description into the section A.14.6]). Since a start code is not preceded with data, it can be deleted by a start code detector.

[0928]The parser events ERR PSPARE and ERR GSPARE direct detection of this information during the H.261 operation. The corresponding events in an MPEG operation period are an ERR EXTRA picture and an ERR EXTRA slice.

[0929]Generation of a parser event will pass through and present the 1st byte of additional information via the register romrevision.

[0930]The continuation state of an animation demultiplexer register determines the action mode after the event was cleared. When holding this register value 0, all the remaining additional information is consumed by the animation demultiplexer, and no event is generated. An event is generated as each byte of additional information reaches an animation demultiplexer, when continuation is set to 1. This continuation is

continued until additional information is used up or continuation is set to 0.
[0931]Notes: The 1st byte of 1 extension / user datum is always shown via a romrevision register regardless of the state of continuation.
2) There is no event which shows that read the byte of the last of extension/user datum was finished.
[0932]A.14.7.1 "Generation of a FIELD INFO token"
During the MPEG operation, when the register field info is set to one, the 1st byte of all extra information pictures is stationed in a FIELD INFO token. This action mode is not covered by the standardization activity of MPEG. The definition of a FIELD INFO token is shown in Table 4 - 11.
[0933]When the field info is set to one, no parser event is generated to the 1st byte of an extra information picture. However, an event is generated to all the succession bytes of an extra information picture. When there is only one single byte of an extra information picture, no parser event is generated.
[0934]A.14.8 "Change in an MPEG sequence layer"
An MPEG sequence header describes the characteristic shown in the next of the animation which is going to be decrypted.
[0935]* When level and the buffer size space decoder of the aspect ratio * picture rate * coding data rate * animation buffer verifier of a vertical size * pixel decrypts a sequence header, When these arbitrary parameters change, a parser event ERR MPEG sequence is generated.
[0936]A.14.8.1 "Change of picture size"
When picture size changes, a user's software reads the value in horizpels and vert vertical pels, and calculates the new value for loading to the registers horizmacroblocks and vertmacroblocks.
[0937]Section A.15 "Space decryption"
Based on this invention, space decryption occurs between the output of token buffers, and the output of a space decoder.
[0938]There are three main units which take the responsibility for space decryption, i.e., a reverse modeler, an inverse quantization device, and a reverse separate cosine transformer. In the input (from token buffers) of this section, a data token includes the run of a coefficient and level expression which were quantized. In an output (reverse DCT), a data token contains 8x8 blocks of pixel information.
[0939]A.15.1 "Reverse modeler"
The data token in token buffers includes the information about the number of the value of the quantized coefficient, and the zero between the displayed coefficients. A reverse modeler expands the information about the run of zero so that each data token may contain the value of 64. In this stage, the value in a data token is the quantized coefficient.
[0940]Regardless of encoding specification present in use, the inverse model-ized

process is the same. The configuration is unnecessary entirely.
[0941]In order to understand much more well all the necessary conditions about modeling and an inverse model-ized function, readers can investigate arbitrary picture coding standards.
[0942]A.15.2 "Inverse quantization device"
in an encoder -- a quantizer -- the resolution of a DCT coefficient -- a DCT output is divided in order to reduce a degree. The function of the inverse quantization device in a decoder is giving multiplication to these quantized DCT coefficients, in order to restore so that the original value may be resembled.
[0943]A.15.2.1 "Outline of a standard quantization plan"
There is an important difference in the quantization plan under which it is used by each of different encoding specification. In order to understand in detail about the quantization plan under which it is used by each standard, readers need to study a related encoding specification document.
[0944]A register iq coding standard constitutes the operation of an inverse quantization device so that the necessary condition of a different standard may be suited. In normal operation, this coding register is automatically loaded by the coding standard token. Please refer to the section A.21.1 about much more information about the configuration of a coding standard.
[0945]The main differences between quantization plans are a number of sauce which carries out multiplication to the quantized coefficient. These are outlined next. Although not explained here, there is a detailed difference in required arithmetic operation (it rounds off **).
[0946]A.15.2.1.1 "General view of H.261 IQ"
In H.261, in order to adjust a coefficient, one single "scale factor" is used. The encoder can change this scale factor periodically, in order to adjust the data rate created. A slightly different rule is applied to the "DC" coefficient within the Intra coding block.
[0947]A.15.2.1.2 "General view of JPEG IQ"
Base-line JPEG permits the picture containing a color component which is different to a maximum of four in each scan. 64 entry-quantity child-ized table can be specified to these four color components. Each entry of these tables can be used as a "scale" factor to one of the coefficients by which 64 was quantized.
[0948]The value for JPEG quantization tables is contained in coding JPEG data, and is automatically loaded to a quantization table.
[0949]A.15.2.1.3 "General view of MPEG IQ"
H.261 and JPEG quantization art are used for MPEG. It is usable in four quantization tables respectively provided with the entry of 64 like the case of JPEG. However, the directions for a table completely differ.
[0950]Two the "types" of data is intra and non-Intra. Different tables to each data

type are used. Two ″default″ tables are defined by MPEG. One is an object for intra data and refer to Tables 130 and 131 which is an object for non-intra data for one now. These default tables must be entered in the quantization table memory of a space decoder, before attaining MPEG decryption.
[0951]Similarly, MPEG permits two ″download″ quantization tables. One is for using it with intra data.
One is for using it with non-intra data now.
The value to these tables is contained in an MPEG data stream, and is loaded to a quantization table memory automatic target.
[0952]The value outputted from a table is corrected by a scale factor.
[0953]A.15.2.2 ″Inverse quantization device register″
[0954]
[Table 129]
In this invention, before a quantization table memory becomes accessible, iq access register must be set. When reading is tried by the table while iq access was set to 0, a quantization table memory returns value zero.
[0955]A.15.2.3 ″Composition of an inverse quantization device″
In normal operation, since it is automatically constituted by the coding standard token, it is not necessary to constitute the encoding specification of an inverse quantization device.
[0956]A quantizer table is not used to H.261 operation. A special configuration is not needed. To JPEG operation, the table needed by the inverse quantization device must be automatically loaded by the information extracted from coded data. MPEG operation needs to load a default quantization table. This loading must be performed while iq access is set to 1. The value in Table 130 must be filled in by 0x3F from the place 0x00 of the extension address space of an inverse quantization device (accessible via a keyhole register iq keyhole address and iq keyhole data). Similarly, the value of Table 131 must be filled in by 0x7F from the place 0x40 of the extension address space of an inverse quantization device.
[0957]
[Table 130]
[0958]
[Table 131]
A.15.2.4 ″List constitution from a token″
Instead of constituting an inverse quantization device table via MPI, it is possible to initialize these inverse quantization device tables by a token. These tokens can be supplied by either the coded dataport or MPI.
[0959]A QUANT table token is described by Table 3 − 11. Said token has the one bit field [two] tt which specifies any of the place of 4 (0−3) ** of a table are defined by the token. It is required to load the default definition of Tables 0 and 1 to MPEG

operation.
[0960]A.15.2.5 "Value of a quantization table"
To both JPEG and MPEG, a quantization table entry is an 8-bit number. The values from 255 to 1 are just. The value 0 is unjust.
[0961]A.15.2.6 "Number order of a quantization table"
The value of a quantization table is used in the "ZIG ZAG" scanning order foreword (refer to encoding specification). It must be considered that a table is a one-dimensional array with the value of 64 (they are not 8x8 arrays). The entry of the table in a low address corresponds to a low frequency DCT coefficient.
[0962]When the value of a quantization table is possessed by the QUANT table token, the 1st value after a token header is a front entry for the "DC" coefficients.
[0963]A.15.2.7 "Inverse quantization device test registers"
[0964]
[Table 132]
A.15.3. "Discrete inverse cosine transform"
The reverse separate conversion processor of this invention suits the necessary condition specified in H.26 l. CCITT recommendation and the IEEE specification P1180, and suits the necessary condition described by the present revised proposal of MPEG.
[0965]The discrete inverse-cosine-transform process is the same regardless of whether which coding standard is used. The configuration by a user is unnecessary entirely.
[0966]There are two events relevant to a reverse separate conversion processor.
[0967]
[Table 133]
In order to understand DCT and a reverse DCT function much more well, readers can investigate arbitrary picture encoding standards.
[0968]Section A.16 "Connection with a space decoder output"
The output of a space decoder is a standard token port of 9-bit-wide data word. Please refer to the section A.4 for the detailed information about the electric action mode of an interface.
[0969]The token in an output is dependent on the encoding specification under adoption. As an example, in this section of this indication, when constituted to JPEG operations, the output of a space decoder is considered. Since a time decoder does not correct the token sequence obtained as a decryption result of JPEG, in this section, it describes the token sequence observed in the output of a time decoder during the JPEG operation.
[0970]However, in both sides, MPEG and H.261 need use of a time decoder. Please refer to the section A.19 for the information about connection with the time decoder output in the case of being constituted MPEG and for H.261 operations. In this section, the circuit design for displaying whether it has set to the output of the space decoder

and the token of a gap can be used and the output concerned is faced, and it is identified whether the token of a gap is the most useful. Although other tokens exist, since those outputs are display needlessness, it does not inquire here. The item intensively examined in this section is shown below.
[0971]* A start of a sequence, and an identifying method of an end.
[0972]* A start of a picture, and an identifying method of an end.
[0973]* An identifying method of when to display a picture.
[0974]* Method A.16.1 of identifying the picture data arrangement place in a display ″Structure of a JPEG picture″
In this section, it surveys about the feature of JPEG syntax. For details, please refer to encoding specification.
[0975]JPEG provides various mechanisms for coding each picture. In order to provide the mechanism which codes an animation, JPEG does not try to describe how to code the assembly of a picture collectively.
[0976]The space decoder based on this invention supports the sequential operating mode as the base line of JPEG. This syntax has three main levels, i.e., an image, a frame, and a scan. One sequential image contains only one single frame. One frame can contain a different image (color) ingredient of a before [from 1 / 256]. In various methods, these image components can be brought together in a scan. Each scan can contain the image component of a before [from 1 / 4] (refer to drawing 90 ″a general view of a sequential structure as the JPEG base line″). When a scan contains one single image component, it is a non-Inta reaved scan, and when two or more image components are included, it is an Inta reaved scan. One frame can contain the mixture of an Inta reaved can and a non-Inta reaved scan. The number of the scans which a frame can include is determined by the limit of 256 about the number of the image components which a frame can contain. In one Inta reaved scan, data is organized per the minimum coding (MCU) similar to the macro block used for MPEG and H.261. In a picture, these MOU(s) are arranged in order of a raster. MCU in a non-Inta reaved scan is 8x8 blocks of one single. Also in this case, raster systematization of the MCU is carried out.
[0977]The space decoder can decrypt easily the JPEG data containing a color component which is [from 1 to 4] different. The file which describes a large number of ingredients can be decrypted similarly. However, in order to accommodate 1 set next to the ingredient decrypted, reconstruction of a certain grade may be required between scans.
[0978]A.16.2 ″Token sequence″
Refer to Table 92 changed into MPEG of the similarity specified as the token by the start code detector, and drawing 91 ″JPEG picture by which tokenization was carried out″ for a JPEG marker code.
[0979]Section A.17 ″Time decoder″

* 30, MH7 operation.
[0980]* Provide the time decryption for MPEG and H.261 animation decoders.
[0981]*H.261 CIF and a QCIF format.
[0982]* It is a maximum of 704x480, 30 Hz, and 4:2:0 whenever [MPEG animation resolution].
[0983]* A flexible chroma sampling format.
[0984]* MPEG picture sequence order repair is possible.
[0985]* Glue loess DRAM interface.
[0986]* Single+5V power supply.
[0987]* 208 pin PQFP package.
[0988]* Maximum dissipation consumption 2.5W.
[0989]* Use standard page mode DRAM.
[0990]A time decoder is a companion chip for a space decoder.
And the time decryption needed by H.261 and MPEG is provided.

[0991]A time decoder performs all the prediction formation functions needed by MPEG and H.261. The time decoder can decrypt CIF and a QCIF H.261 animation using one single 4Mb DRAM (for example, 512kx8). If 8MbDRAM (for example, two 256kx16) is used, 704x480, 30 Hz, and a 4:2:0MPEG animation can be decrypted.
[0992]The Intra coding plan (for example, JPEG) does not need time decoder **. When a time decoder is contained in a multiplex standard decoder, the time decoder concerned supplies a decrypted JPEG picture to the output.
Notes: The above-mentioned value only shows an example of one example of this invention, and does not necessarily mean restriction. It should be understood that it is possible to use other values and ranges without deviating from the range of this invention.
[0993]A.17.1 "Time decoder signal"
[0994]
[Table 134]
[0995]
[Table 135]
[0996]
[Table 136]
[0997]
[Table 137]
[0998]
[Table 138]
A.17.1.1 "Connectionless pin "nc""
The pin by which the label indication is carried out to "nc" in Table 138 is a pin which is not used for present this invention but is reserved for the future product. It must

leave these pins without connecting. Don't connect these pins with VDD, GND, and all the signals of mutual or others.
[0999]"A.17.1.2 VDD and GND pin]
All the VDD(s) and GND pins must be connected to an applicable power supply so that I may naturally be understood. If all VDD(s) and GND pins are not used correctly, the device does not operate correctly.
[1000]A.17.1.3 [Connection of the test pin for a normal system operation]
Nine pins of a time decoder are reserved as an object for internal tests.
[1001]
[Table 139]
A.17.1.4 [JTAG pin for a normal system operation]
Refer to section A.8.1.
[1002]
[Table 140]
[1003]
[Table 141]
[1004]
[Table 142]
[1005]
[Table 143]
[1006]
[Table 144]
[1007]
[Table 145]
[1008]
[Table 146]
[1009]
[Table 147]
[1010]
[Table 148]
Section A.18 "Time decoder operation"
A.18.1 "Data input"
The input dataport of a time decoder is a token port of a standard with the data word of 9 bit width. In almost all applications, direct continuation of this is carried out to the output token port of a space decoder. Refer to the section A.4 for the detailed information about the electric act of this interface.
[1011]A.18.2 "Automatic formation"
The parameter about the coded video picture format is automatically loaded to the register in a time decoder by the token generated by a space decoder.
[1012]

[Table 149]
A.18.3 ″Manual formation″
The user has to form an application (passing microprocessor interface) dependent.
[1013]A.18.3.1 Do [″when it forms″]?
The time decoder should be formed only when data processing is not performed. This is the default State after reset was canceled. By writing 1 in a chip access register, it stops and the reconstitution of the time decoder can be carried out. After formation is completed, 0 must be written in chip access.
[1014]Refer to the section A.5.3 for the details about when a DRAM interface is formed.
[1015]A.18.3.2 ″DRAM interface″
Before the timing of a DRAM interface can decode the video (for example, H.162 or MPEG) coded in prediction, it must be formed. Refer to the section A.5 ″DRAM interface.″
[1016]
[Table 150]
[1017]
[Table 151]
A.18.3.3 The ″number of picture buffer registers″
A picture buffer pointer (18 bits) and an ingredient offset (17 bits) register specify not a byte address but a block (8x8 bytes) address.
[1018]A.18.3.4 ″Picture buffer store assignment″
In order to decode the video (H.261 or MPEG) coded in prediction, the time decoder must process two picture buffers. Refer to the section A.18.4 and A.18.4.4 for the details about how these buffers are used.
[1019]In order that a user may memorize the single picture of required format video (without piling up with other picture buffers), It must confirm that there is memory sufficient on [each] a picture buffer pointer (picture buffer 0 and picture buffer 1). Usually, one of the picture buffer pointers is set as 0 (that is, pars basilaris ossis occipitalis of memory), and it is set up as the middle of a memory space pointed out in other one.
[1020]A.18.3.4.1 The ″usual formation for MPEG or H.261″
Both H.261 and MPEG use the ratio 4:1:1 (it is got blocked and there is a 4 times as many luminosity pixel as the pixel in one of chrominance components) between different color components.
[1021]A. The ingredient 0 is a brightness component and the ingredients 1 and 2 are chrominance as documented by 3.5.1 and an ″ingredient specified number.″
[1022]The example of formation of an ingredient offset register sets component offset 0 as 0 so that the ingredient 0 may start in a picture buffer pointer. Similarly, component offset 1 can be set as four sixths of picture buffer size, and component

offset 2 can be set as five sixths of picture buffer size.
[1023]A.18.3.5 "Picture sequence rearrangement (re-ordering)"
: for which MPEG uses three different picture types -- intra -- (I), expected (P), and (B) interpolated in for two way types. B picture -- two picture: -- it is based on the anticipation from the thing from the future, and the thing from the past. Before the decipherment of B picture is called for, an order of a picture is corrected with an encoder so that it may be decoded from the data in which I and P picture were coded.
[1024]A picture sequence must be corrected before these pictures are displayed. The time decoder can provide this (it is set as register MPEG reordering=1 more) picture rearrangement. Or a user may wish that he would like to perform a picture rearrangement as a part of his display interface function. The video resolution which may be decoded may be reduced by forming a time decoder so that a picture rearrangement may be provided. A. Refer to 18.5.
[1025]A.18.4 "Prediction formation"
The requirements for prediction formation of H.261 decoding and MPEG decoding completely differ. A coding standard token is automatically formed so that the requirements for prediction of the standard from which a time decoder differs may be accommodated.
[1026]A.18.4.1 "JPEG operation"
Since JPEG does not need time decoding if formed in JPEG operations, prediction is not performed.
[1027]A.18.4.2 "H.261 operation"
In H.261, prediction is performed only from the picture by which decoding was carried out. A motion vector is only specified as the accuracy of an integer pixel as conditions. An encoder can be specified that a low-pass filter is applied to a prediction result.
[1028]It is written in the picture buffer in off-chip DRAM so that it can be used, when carrying out decoding of the following picture, as decoding of each picture is carried out. The picture by which decoding was carried out appears in the output of a time decoder as it is written in off-chip DRAM.
[1029]A readout system is turned to an H.261 standard about the details and the related operation operation about prediction. The time decoder of this invention has observed the requirements for H.261 thoroughly.
[1030]A.18.4.3 MPEG operation (not accompanied by a rearrangement)
The operation of a time decoder changes to each of three different MPEG picture types (I, P, B). Although I picture does not need the further decoding by a time decoder, since it is used when carrying out decoding of P and the B picture to behind, a picture buffer (frame storage) must memorize.
[1031]It is necessary to form in decoding of P picture the prediction from P or I picture by which decoding was carried out before. Since decoded P picture uses it when it decodes P and B picture, it is memorized in a picture buffer. MPEG is

specified as the accuracy whose motion vector is half a pixel. A filter on chip provides the interpolation for supporting this half a pixel accuracy.
[1032]B picture can need the prediction from both picture buffers. Half a pixel motion vector resolution accuracy requires interpolation of picture data on chip like P picture. B picture is not memorized by the off-chip buffer. It is temporary in them.
[1033]All the pictures appear in the output port of a time decoder as those numerals are decoded. Therefore, the picture sequence is the same as the thing in the coded MPEG data (see the upper portion of drawing 94).
[1034]Refer to the draft of the MPEG standard proposed for the details and the related operation operation about prediction. These requirements are satisfied by the time decoder of this invention.
[1035]A.18.4.4 "MPEG operation (accompanied by a rearrangement)"
If formed for the MPEG operation (MPEG reordering=1) accompanied by a picture rearrangement, prediction formation operation is the same in the above-mentioned section A.18.4.3. However, in order to carry out the rearrangement of the picture sequence, additional data communications are carried out.
[1036]B picture decoding is as having described in the section A.18.4.3. However, I and P picture are not outputted when decoding is carried out. Instead, it is written in an off-(as mentioned above) chip buffer, and it is read only when the following I or P picture arrives for decoding. A.18.4.4.1 "Decoder start-up characteristic" The output of the first I picture is delayed until decoding of the following P (or I) picture starts. This must be taken into consideration when estimating the start-up characteristic of a video decoder.
[1037]A.18.4.4.2 "Decoder stop characteristics"
A time decoder is dependent on the following P or I picture, in order to carry out the flash plate of the front picture from the off-chip buffer (frame storage). When this starts the time of the end of a video sequence, and a new video sequence, a result appears. A space decoder provides the facilities for making a "fake" I/P picture at the time of the end of a video sequence, in order to carry out the flash plate of the last P (or I) picture. However, if the following video sequence starts, the flash plate of the picture of this "imitation" will be carried out.
[1038]A space decoder provides the option for controlling the picture of this "imitation." This is useful when it is known that a new video sequence will be promptly supplied to a decoder after completion of the old sequence. The picture of the beginning within this new sequence will carry out the flash plate of the picture of the last of a front sequence.
[1039]A.18.5 "Video resolution"
When carrying out decoding of the MPEG, the video resolution which can support a time decoder is restricted by the memory band width of the DRAM interface. For MPEG, they are a case where it is accompanied by :MPEG picture rearrangement

which needs to take two cases into consideration, and a case where it does not follow.
[1040]The section A.18.5.2 and A.18.5.3 are discussing the requirements in the worst case of being required by the present draft of MPEG agreement. The subset of MPEG with the requirements for lower memory band width is planned. For example, the requirements for memory band width are reduced seriously, without instead using B picture, using only an integer solution image degree motion vector. Analysis of this **** subset is not conducted here.
[1041]A.18.5.1 [Characteristic of a DRAM interface] in order to cross a DRAM interface and to transmit data. Timing formation and data bus width of the DRAM interface for fitting the number of cycles taken to : depending on the number of factors, and DRAM used (8, 16, or 32 bits)
- Refer to the section A.5 and a "DRAM interface" about the information about detailed formation of a DRAM interface for the prediction to - integer pixel accuracy for the prediction to data-communications type:, 8x8-block read-out, or writing and half a pixel accuracy.
[1042]It is shown how many DRAM interfaces "cycle" are required for Table 152 because of each type of data communications.
[1043]
[Table 152]
Table 154 takes the number of Table 152 and evaluates them for "typical" DRAM. In this example, it is assumed that they are 27 MHz clocks. Although 27 MHz is used here, it should recognize that it is not what is restricted to it. Eleven ticks (102 ns) are required for an access start, and six ticks (56 ns) are required for data communications.
[1044]A.18.5.2 "MPEG resolution without a rearrangement"
Loading of peak memory band width is generated when carrying out decoding of the B picture. In the scenario in "the case of being the worst", the B frame is formed from the prediction from both picture buffers, and all the prediction is against half a pixel accuracy.
[1045]
[Table 153]
If the example of a number from Table 153 is used, in order to read data required for two (passing the interface of 32 bit width) exact half a pixel accuracy prediction, it turns out that DRAM interface 3815ns is required. The resolution which can support a time decoder is determined by these numbers of prediction executable within 1 picture time. In this example, the time decoder can process 8x8 blocks of 8737 to one 33-ms picture cycle (for example, for 30-Hz video).
[1046]If required format video is 704x480, each picture includes 8x8 blocks of 7920 (if the chroma sampling of 4:2:0 is taken into consideration). This format video consumes about 91% of the DRAM interface bandwidth which can be used (before taking other

factors, such as DRAM refresh, into consideration). Therefore, the time decoder can support this format video.
[1047]A.18.5.3 "MPEG resolution accompanied by a rearrangement"
If an MPEG picture rearrangement is used, while decoding of the P picture is carried out, the scenario in the case of being the worst is encountered. P before - which returns :, the gestalt prediction, and the result by which three loading is performed on a DRAM interface between this time, or I picture is read.
[1048]when the interface of 32 bit width can be used using the example of a number from Table 152, the number of times required for work of these each can be found out. While 1907 second/n is required for prediction formation, 991 ns is respectively required for read-out and writing, and 3889 ns is needed on the whole. A time decoder enables it for this to process 8x8 blocks of 8485 in between in a cycle of 33 ms.
[1049]Therefore, in order to process 704x480 video, about 93% of the memory band width which can be used (refreshment disregarded) will be used.
[1050]A.18.5.4 "H.261"
H.261 only supports two picture formats, CIF (352x288) and QCIF (172x144), at the rate of a picture up to 30 Hz. A CIF picture includes 8x8 blocks of 2376. Uniquely required memory operations are 8x8-block wiring and prediction formation by an integer accuracy motion vector.
[1051]In order to write in each block using the example of a number from Table 153 for the memory interface of 8 bit width, while 3657 ns is needed, it turns out that 3963 second/n is required for 1 block of prediction formation, and 7620 ns is required for every block on the whole. The processing time for one CIF picture is about 18 ms, and has become less than 33 ms required in order to support 30-Hz video considerably.
[1052]A.18.5.5 "JPEG"
The resolution of the JPEG video which can be supported is determined by the space decoder of an invention, or the capability of a display interface. A time decoder does not affect JPEG resolution.
[1053]A.18.6 "Event and error"
A.18.6.1 "Tips top"
In this invention, if 1 is written in chip access, it will be urged for a time decoder to suspend operation and for the reconstitution to be made to be made. Once it is accepted, a time decoder will usually continue operation until it reaches the end of the present video sequence. Then, it is stopped by the time decoder.
[1054]A stop of a chip will generate a chip stop event. If it is chip stopped mask=1, an interrupt will occur.
[1055]A.18.6.2 "Count error"
The time decoder of this invention possesses the adder which adds the prediction to error data. When a difference is between the number of error data bytes, and the number of prediction data bytes, a count error event occurs.

[1056]In the case of count error mask=1, an interrupt occurs and prediction formation stops.
[1057]If one is written in count error event, an event will be cleared and a time decoder will be made to advance. The DATA token which generated the error after that continues. However, probably, the DATA token which generated the error is not a thing of right length (46 bytes). This will generate the further problem easily. Thus, the count error should be produced only when a serious hardware error occurs.
[1058]Section A.19 ″Connection with the output of a time decoder″
The output of a time decoder is a token port of a standard with the data word of 8 bit width. Refer to the section A.4 for the more detailed information about the electric act of an interface.
[1059]The token which exists in the output of a time decoder will be based depending on the coding standard used on whether the rearrangement of the picture is carried out in the case of MPEG. It is specified whether which is the most useful when this section designs the circuit for displaying which token can be used in the output of a time decoder, and its output. Although other tokens probably exist, since it is not necessary to display an output, it does not discuss about them here.
[1060]. [how this section can specify the start and end of : and a sequence which take up the following points preponderantly and discuss them, and] - How is it specified how it is specified how a start and end of a picture can be specified, or when - picture is displayed, or where [of a display] - picture data should be placed?
[1061]A.19.1 ″JPEG output″
At the time of decoding of JPEG data, the token sequence outputted by the time decoder is the same as that of what is seen in the output of a space decoder. I want you to remember that JPEG does not need processing by a time decoder. however, a value negative (it produces from the arithmetic accuracy to which IDCT in a space decoder was limited) in a time decoder sake -- intra -- a data token is investigated and they are transposed to 0.
[1062]Refer to the section A.16 for the detailed argument about the output sequence observed during JPEG operation.
[1063]A.19.2 ″H.261 output″
A.19.2.1 ″A start and end of a session″
H.261 does not sign a start and end of the video stream in a video data. Nevertheless, this is suggested by application. For example, if electrical communication is connected, a sequence will start, and if a line is intercepted, it will end. Thus, in video syntax, the highest layer is a ″picture layer.″
[1064]As for the start code detector of the space decoder by this invention, a sequence start and a coding standard token are automatically inserted before the first picture start. Refer to the section A.11.7.3 and A.11.7.4.
[1065]The user has to insert a flash plate token after the end of coding data in the

end of an H.261 session (for example, when a line is intercepted). In order that : and it with many effects (A. refer to 31.1) may sign this in the end of the last picture, it guarantees that a picture end is emitted.
[1066]- It guarantees that the end of coding data is pushed through a decoder.
A.19.2.2 ″Acquisition of a picture″
Each picture comprises the hierarchy of an element called a syntax layer. When carrying out decoding of H.261, the sequence of the token in the output of a time decoder reflects this structure.
[1067]A.19.2.1 ″Picture layer″
A picture start token is placed before each picture, and a picture and a token continue just after a picture. Naturally H.261 does not contain a picture end. This token is automatically inserted by the start code detector of a space decoder. After the picture start token, the time standard token and the picture type token will continue. A time standard token supports 10 numbers of bits which direct when a picture should be displayed (only 5LSB is used in H.261 among those). Since the H.261 encoder can omit a picture from a sequence (in order to attain a low data rate), this point should be studied by the display system. The abbreviation of a picture may be detected by the time base which only one or more parts between continuous pictures increase.
[1068]Next, a picture type token has the information about a picture format. The display system can investigate this information and it can be detected whether decoding of CIF or the QCIF picture is carried out. However, the information about a picture format can also be used by investigating the register in the Huffman decoder. <Xref: Huffman decoder section >A.19.2.2.2 ″Group of a block layer″
H.each 261 picture is constituted by many ″block groups.″ Before [each] that, a slice (pulled out from group No. [of H.261] and group start code) start token is placed. 8 bit values which direct where [of a display] this token should put the group of a block are owned. This provides the opportunity for the back decoder of a data error to have simultaneity again. If it has the picture area which does not need additional information in order to describe a picture, it provides an encoder with the mechanism which jumps over a block. Since this information is already used in order to ensure that each picture has those blocks in a right location including the block of the number of the rights as for a space decoder and a time decoder, that information is [by] effectively redundant when a slice start reaches the output of a time decoder. In this way, it becomes possible to calculate where the data block outputted by the time decoder should be put by counting the block count outputted since the start of a picture.
[1069]The number supported by slice start is a number with less one than the group of the block count of H.261 (refer to an H.261 standard for detailed information). Drawing 103 shows positioning of the H.261 block group in CIF and a QCIF picture.
[1070]Notes: In this invention, illustrated block numbering is the same as what is

supported by slice start. This differs from the H.261 regulation for carrying out numbering of these groups.
[1071](Each block group's start is shown) Other tokens may be between a slice start and the first macro block. Since they do not need to display picture data, they can be disregarded.
[1072]A.19.2.2.3 ″Macro block layer″
The sequence of the macro block in each group of a block is limited by H.261. There is no special token information explaining the position of each macro block. The user has to count through a macro block sequence, in order to determine where each information is displayed.
[1073]Drawing 105 shows the sequence arranged at each group of a block of a macro block.
[1074]Each macro block includes six data tokens. The sequence of the data token contained in each six group is limited by the macro block structure of H.261. Each data token should contain 64 data bytes correctly because of the 8x8-pixel area of one color component. A color component is contained in the 2-bit number in a data token (refer to section A.3.5.1). However, the sequence of the color component in H.261 is limited.
[1075]Before each group of a data token, many tokens which transmit the information about a motion vector, a quantizer scale factor, etc. are placed. Since a picture does not need to be made to be displayed, these tokens can be disregarded.
[1076]Each data token includes 64 data bytes for one color component of 8x8. These are in a raster state.
[1077]A.19.3 ″MPEG output″
MPEG contains many layers in the syntax. These materialize the concept of a video sequence, a picture group, etc.
[1078]A.19.3.1 ″MPEG sequence layer″
Although the sequence can have a multiplex entry point (sequence start), it should have only one exit point (sequence end). When an MPEG sequence header code is decoded, a space decoder produces the sequence start token following a coding standard token and it.
[1079]Probably, there are many tokens of the sequence header information which describes format video etc. after a sequence start. Refer to Table 3 - 11 for the information about how this data is changed into a token with reference to the MPEG standard of a draft about the information signed in a sequence header. This information that describes format video can be used also in the register in the Huffman decoder.
[1080]This sequence header information may be generated several times, if that sequence has some entry points in the MPEG sequence.
[1081]A.19.3.2 ″Picture layer group″

The MPEG group of a picture provides with the ″entry″ point of a different type what it is at the sequence start time and is provided. A sequence header provides the information about a picture/format video. Therefore, when a decoder does not have the knowledge of the format video used in a sequence, it must start at a sequence start. However, even if it is a position of which group's picture, it should enable it to start decoding, once format video is constituted in a decoder.
[1082]MPEG does not restrict the number of pictures in a group. However, in many applications, since one group provides the reason particle size of random access, he is equivalent to about 0.5 second.
[1083]A picture group's start is directed by the group start token. The header information provided after a group start contains two useful token:TIME CODE and BROKEN CLOSED.
[1084]TIME CODE owns the subset of SMPTE time code information. When this synchronizes a video decoder with other signals, it may be useful. BROKEN CLOSED owns closed gap and the brokenlink bit of MPEG. Refer to the section A.19.3.8 about decoding of the video sequence into which random access was connoted and edited.
A.19.3.3 a picture layer -- the start of a new picture is directed by the picture start token. After this token, a time standard token and a picture type token are. When it is not constituted so that a time decoder may provide a picture rearrangement, temporary standard information may be useful. Especially the information on a picture type may be useful when a display system wants to process B picture at the start of open GOP (A.section 19.3.8 reference).
[1085]Each picture comprises many slices.
[1086]A.19.3.4 ″Slice layer″
The section A.19.2.2.2 is discussed about the block group used in H.261. The slice in MPEG achieves the same function. However, slice structure is not fixed by the standard. 8 bit values which a slice start token owns are values with less one than the ″slice vertical position″ which MPEG transmits. Refer to an MPEG standard draft about explanation of a slice layer.
[1087]Since the space decoder and the time decoder have already used this information in order to ensure that each picture includes the block of the number of the rights in the right location, that information is [by] effectively redundant when a slice start reaches the output of a time decoder. In this way, it becomes possible to calculate where the data block outputted by the time decoder should be put by counting the block count outputted since the start of a picture.
[1088]Refer to the section A.19.3.7 for the argument on the effect which uses an MPEG picture rearrangement.
[1089]A.19.3.5 ″Macro block layer″
Each macro block contains 6 blocks. These appear in the output of a time decoder in the state of a raster (it is clearly written by the draft of MPEG agreement like).

[1090]A.19.3.6 ″Block layer″
Each macro block includes six data tokens. The sequence of the data token contained in each six group is limited by the MPEG agreement draft (this is the same as the macro block structure of H.261). Each data token should contain 64 data bytes correctly because of the 8x8-pixel area of one color component. A color component is contained in the 2-bit number in a data token (A. 3.5.1 references). However, the sequence of the color component in MPEG is limited.
[1091]Before each group of a data token, many tokens which transmit the information about a motion vector, a quantizer scale factor, etc. are placed. Since a picture does not need to be made to be displayed, these tokens can be disregarded.
[1092]A.19.3.7 ″Effect of an MPEG picture rearrangement″
as it explained in A.18.3.5, a time decoder provides an MPEG picture rearrangement -- it can constitute like (MPEG reordering=1). As for the output of P and I picture, the following P/I picture in a data stream is delayed until decoding starts by a time decoder. At the time of the output of a time decoder, the data token of the P/I picture by which decoding was newly carried out is replaced by the data token from an old P/I picture.
[1093]The picture start of a picture, a time standard, and the token of a picture type are temporarily memorized on chip as a picture is written in an off-chip picture buffer, when carrying out the rearrangement of the P/I picture. If read for a display of a picture, these memorized tokens will be searched. Therefore, the P/I picture by which the rearrangement was carried out has the right value for a picture start, a time standard, and a picture type.
[1094]The reorder of none of other tokens under a picture layer is carried out. It picks up the non-data token of a level with a lower picture to the extent that decoding was carried out now which has passed as it is read for a display of the P/I picture by which the rearrangement was carried out. Thus, the token of these sub picture layers should be disregarded.
[1095]A.19.3.8 ″Random access and edited sequence″
A space decoder provides the equipment which helps right video decoding after the random access to an edited MPEG video data and an MPEG video data.
A.19.3.8.1 ″Open GOP″
The picture group (GOP) can start by B picture predicted from P picture in front GOP. This is called ″open GOP.″ Drawing 116 is illustrating this. The pictures 17 and 18 are B pictures in the start of the 2nd GOP. In an encoder, GOP can code these two pictures using the prediction from the P picture 16 and the I picture 19, saying ″It is opened.″ Or instead, the encoder could be restricted so that the prediction only from the I picture 19 may be used. In this case, the 2nd GOP is called ″the closing GOP.″
[1096]Since the GOP has already carried out decoding of the P picture 16 if a decoder starts video decoding in the first GOP, even if it obtains with the GOP and is

open, and it encounters the 2nd GOP, any problem does not arise? However, if they are dependent on P16 when a decoder performs random access and starts decoding in the 2nd GOP, it cannot carry out decoding of B17 and B18 (if it is got blocked and GOP is opened).
[1097]If the space decoder of this invention encounters open GOP as first GOP following reset or a flash plate token is received, it will be assumed that the random access to open GOP occurred. In this case, the Huffman decoder will consume data by the usual method for B picture. However, it will output B picture predicted by the motion vector (0, 0) which is separated from I picture. As a result, the pictures (it can set for the above-mentioned example) B17 and B18 will become the same as that of I19.
[1098]This act ensures right maintenance of the VBV rule of MPEG. It ensures that B picture exists in the output in the position in the output stream expected by other data channels. For example, an MPEG system layer provides a video data with the display time information about audio information. The stamp of video presentation time directs the first display picture in GOP, i.e., the picture of the time base 0. In the above-mentioned example, the picture displayed on the beginning is B17 after the random access to the 2nd GOP.
[1099]A broken closed token owns a MPEG closed gop bit. Therefore, in the output of a time decoder, the output of B picture is a genuine article, or it is possible to judge whether the ″substitute″ was introduced by the space decoder. He may wish for a certain application to lecture on special treatment, when the picture of these ″substitutes″ exists.
[1100]A.19.3.8.2 When edited video application edits an MPEG video sequence, it may destroy the relation between two GOP(s). If GOP after edit is open GOP, carrying out decoding of the B picture correctly in the beginning of GOP will not be able to do it any longer. The application which edits MPEG data can set up a broken link bit in GOP after edit, and can direct that decoding of these B pictures cannot be carried out to a decoder.
[1101]The Huffman decoder will carry out decoding of the data by the usual method for B picture, when GOP with the link with which the space decoder was destroyed is encountered. However, it will output B picture predicted by the motion vector (0, 0) which is separated from I picture. As a result, the pictures (it can set for the above-mentioned example) B17 and B18 will become the same as that of I19.
[1102]A broken closed token owns the broken link bit of MPEG. Therefore, in the output of a time decoder, it is possible to judge whether the output of B picture is a genuine article, or it is the ″substitute″ introduced by the space decoder. He may wish for a certain application to lecture on special treatment, when the picture of these ″substitutes″ exists.
Section A.20 ″Postscript lump DRAM interface″

The detailed thia ming of :interface which interfaces are two methods and can be constituted can be constituted so that various different DRAM types may be accommodated.
[1103]"Width" of a DRAM interface can be constituted so that the trade-off of expense/performance may be provided.
[1104]
[Table 154]
[1105]
[Table 155]
[1106]
[Table 156]
[1107]
[Table 157]
[1108]
[Table 158]
[1109]
[Table 159]
A.20.1 "Interface timing (tick)"
In this invention, DRAM interface timing is pulled out from the clock (decoder clock) which runs by 4 times of the rate of an input clock of a device. This clock is made by PLL on chip.
[1110]Since it is brief, the cycle of this high-speed clock is called a tick.
[1111]A.20.2 "Interface operation""
An interface uses the fast page mode of DRAM. read-out of :, read-out, writing, and each refreshment or write-in access by which access of three different types is supported transmits 1-64 bytes of burst in one DRAM page address. Read-out and write-in transmission are not mixed in one access. Each continuous access is processed as random access to a new DRAM page.
[1112]A.20.3 "Access structure"
: and access start-data transmission each access from which each access comprises two portions are started by access start, and the data-communications cycle beyond one or it continues after that. Modification of an access start and read-out of both data-communications cycle, writing, and refreshment occurs.
[1113]An interface is still this state in the end of the data communications of the last in 1 access until it makes that new will be in a default state and is started. When the last access is completed, shortly after it is [start] ready for new access, the new access is started.
[1114]A.20.3.1 "Access start"
An access start provides read-out or write-in transmission with a page address, and sets up some initial signal states. when it is : and lead start light start refreshment

starts of each with three different access starts, the timing and the row address of /RAS/are controlled by the registers RAS falling and page start length. The state of /OE/and DRAM data [31:0] is held until it becomes /RAS/from the end time of earlier data transmission. When the access start of three different types becomes /RAS/, the methods of making /OE/and DRAM data [31:0] drive only differ. Refer to Drawing 118.
[1115]
[Table 160]
A.20.3.2 "Data communications"
Only one refresh cycle follows the next of :, a high-speed page read cycle, and a high-speed page REITO write cycle refresh cycle refreshment start in which three different types of data-communications cycles are. The high-speed page lead (or light) cycle beyond one or it follows the next of a lead (or light) start.
[1116]In a read cycle start, CAS is made into a high and a new column address drives.
[1117]A REITO write cycle is used. /WE/is made into a low only back 1 tick of /CAS/. As for output data, the back 1 tick drive of the address is carried out.
[1118]As /CAS/in front of /RAS/, a refresh cycle begins by a refreshment cycle start, and, in the inside of a refresh cycle, any interface signal activities cannot be found. The purpose of a refresh cycle is to fill the minimum /RAS / low cycle that DRAM needs.
[1119]A.20.3.3 "Interface default state"
: by which an interface signal goes into a default state at the time of the end of access -- A.20.4 which has been stabilized as for -addr high - data and whose /OE/of -/RAS/, /CAS/, and /WE/are still the states before it "Data bus width"
As for the 2-bit register DRAM data width, the width of the data path of a DRAM interface is constituted. This makes DRAM expense the minimum, when working in a small picture format.
[1120]
[Table 161]
A.20.5 "Address bit"
A 24-bit address on chip is made. How it is used in order that this address may form a low and a column address changes with numbers of bits chosen for data bus width and a row address. no use of an internal address bit is allowed with a certain composition (therefore -- "-- it hides and bit" is made).
[1121]A row address is pulled out from the omitted portion of an address. This makes the maximum the rate for which DRAM is refreshed automatically.
[1122]A.20.5.1 "Column address bit of a low order"
In order to provide the address for fast page mode transmission up to 64 bytes, the 4-6-bit least important column address is used. The number of address bits required in order to control these transmissions changes with width of a data bus. (A. Refer to

20.4)
A.20.5.2 "Row address bit"
The number of bits taken from the intermediate section of a 24-bit internal address in order to provide a row address is formed with the register row address bits.
[1123]
[Table 162]
The width of the row address used differs by whether it is the off-chip with which decoding of [for the type of DRAM used and MSB of a row address to access the multiplex bank of DRAM] was carried out.
[1124]Notes: A row address is pulled out from the middle of an internal address. In order that a bit with a row address may choose DRAM banks, when decoding is carried out, all the possible value of these "bank selection bits" must choose DRAM banks. Otherwise, a hole will be left behind to an address space.
[1125]
[Table 163]
A.20.6 "-izing which can be DRAM interfaced (enable)"
There are two kinds of methods in making all the output signals on a DRAM interface into high impedance. They are a DRAM enable register and a DRAM enable signal. It must be the logic 1 in order that a DRAM interface may operate a register and a signal. If either is low, an interface will be made into high impedance and the data communications through an interface will be suspended.
[1126]The capability to have a DRAM interface in high impedance, When the space decoder (or time decoder) is not used, it is provided in order that other devices can test or use DRAM controlled by a space decoder (or time decoder), It is not for other devices to be able to share a memory during the usual operation.
[1127]A.20.7 "Refreshment"
Unless it becomes impossible by writing in the register no refresh, a DRAM interface is an interval determined with the register refresh interval, REFURESSHU [/CAS/in front of /RAS / refresh cycle is used, and / DRAM] automatically.
[1128]The value of refresh interval specifies the interval between the refresh cycles in the period of 16decoder clock cycle. The value of the range of 1-255 can be chosen. The value 0 is automatically loaded after reset, and it is forced so that a DRAM interface may perform a refresh cycle continuously, until an effective refresh interval will be formed (once it is enabling). It is recommended that refresh interval is formed only once [each / after-reset].
[1129]A.20.8 "Signal strength"
The driving force of the output of a DRAM interface is formed of a user using 3 bit registers / CAS / strength, /RAS / strength, and addr strength. MSB of this triplet value chooses either of the rates of edge of a high speed or a low speed. The output for the load capacitance from which a bit with two low importance differs is formed.

[1130]The default intensity after reset is 6, and when loaded at 12 pF, it forms the output which takes about 10 ns to the driving signal between GND and VDD.
[1131]
[Table 164]
It will fulfill AC electrical property specified in Table 168 - 169, when an output is about formed for loading under drive. Probably each output suits the load mostly, therefore the minimum overshooting will generate it after signal transition, when formed appropriately.
[1132]A.20.9 ″After reset″
All DRAM interface formation registers are reset by those default value after reset. : which describes the most important thing below among these default composition -- it is allowed to make -DRAM interface impossible (disable) and to go to high impedance.
[1133]- A refresh interval is formed in the special value 0, and it means execution of a continuous refresh cycle, after an interface is made re-possible.
[1134]- A DRAM interface is set as formation of the maximum low speed.
[1135]Electric power is impressed first, and almost all DRAMs require a ″pause″ for 100 microseconds - 500 microseconds, after many refresh cycles continue before operation becomes possible after that usually.
[1136]After reset, promptly, a DRAM interface is inactivated until both a DRAM enable signal and a DRAM enable register are set up. The DRAM interface will perform the refresh cycle (it responds to the clock frequency used and is every about 400 ns) until a DRAM interface is formed, when these are set up.
[1137]In order to confirm having generated before a required number of refresh cycles after power up tried data communications, a user has the responsibility for ensuring a ″pause″ of DRAM, as there is sufficient time after making a DRAM interface possible.
[1138]While declaring reset, a DRAM interface cannot be refreshed for DRAM. However, the reset time which a decoder chip requires can reset them, and before the contents of the DRAM collapse after that, it is short enough to such an extent that it can make a DRAM interface re-possible. This may be needed during debugging.
[1139]
[Table 165]
[1140]
[Table 166]
[1141]
[Table 167]
A.20.10.1 ″AC characteristics″
[1142]
[Table 168]
[1143]

[Table 169]

Section B.1 ″Start code detector″

B.1.1 ″View″

As it mentioned above in drawing 18, a start code detector (SCD) is the block of the beginning on a space decoder. The fundamental purpose is to detect the start code of MPEG, JPEG, and H.261 in an input data stream, and to replace them with a related token. It enables user access to an input data stream via a microprocessor interface further, and carries out preliminary formatting and ″order″ of a token data stream. I want you to remember that SCD can receive either of the data already assembled by raw byte data or token format.

[1144]Typically, start codes are 24, 16, and 8 bits respectively to MPEG, H.261, and JPEG. A start code detector receives input data from a byte from either a microprocessor interface (upi), or a token / byte port, and moves it through three shift registers. The 1st register is 8-bit parallel series out, and the 2nd register is programmable length (16 or 24 bits), It is a place where a start code is detected, and the 3rd register is 15 bit width, and it is used in order to carry out reformatting of the data to a 15-bit token. There are two ″tag″ shift registers (SR) which move to the 2nd and 3rd SRs and parallel. The tag for directing whether as for these, it is bad whether the bit which unites within data SR is good is included. The input byte who is not a part of data token and is not recognized by SCD enables it to bypass a shift register, and the flash plate of all the three shift registers is carried out (becoming empty), and it is outputted when the contents are outputted successfully. The recognized non-data token is used in order to form SCD, a spring trap, and a set flag. They bypass a shift register and are outputted, without being changed.

[1145]B.1.2 ″Main block″

The hardware for a start code detector comprises ten State machines.

[1146]B.1.2.1 ″Input circuit (scdipc.sch.iplm.M)″

The operation mode the number of input circuits is [operation mode] three: It has a token, a byte, and a microprocessor interface. These modes are inputted by the user through upi as a token stream as a byte stream (although it is, 2-wire system interface is used) thing raw in data. In all the cases, an input circuit always outputs a right data token by generating a data token header in a suitable case. The transition from/to upi mode is synchronized by the system clock, and upi is kept waiting to the safe point in the data stream before obtaining access. It is determined whether byte mode pins are whether an input circuit is in token mode, and a byte mode. The initial notice (therefore, a coding standard token is emitted) to the system about which standard is decoded may be performed in either of the 3 modes.

B.1.2.2 ″Token decoder (scdipnew.sch, scdipnem.M)″

This block carries out decoding of the input token, and emits a command to other blocks.

[1147]

[Table 170]

Notes: The change in a coding standard is sent to all the blocks via a 2 line interface, after the flash plate of the SR is carried out. This ensures that change from one data stream to another data stream occurs on a right point all the time among SCD. This principle is applied all the time during a display so that it can flow through all the chips before a stream with a new change in a coding standard.

[1148]B.1.2.3 "JPEG (scdjpeg.sch, scdjpegm.M)"

The start codes (marker) in JPEG differ enough, and all JPEG has the State machine to self. In this invention, this block processes JPEG marker detection, the counts/checks of length, and all the removal of data. A flag indication of the detected JPEG marker is given as a start code (refer to the below-mentioned text with v nott), and the override of the command from scdipnew is carried out, and it is forced it so that it may bypass. Operation is best explained by the code.

```
switch(state)
{case(LOOKING): if(input=0xff)
{state=GETVALUE;/*Found a marker*/ remove;/*Marker gets removed*/}
state=LOOKING;
break;
case(GETVALUE); if(input=0xff)
{state=GETVALUE;/*Overlapping markers*/ remove;
}
else if(input=0x0-0
{state=LOOKING;/*Wasn't a marker/ insert(0xff);/Put the 0xff back/}
else{command=BYPASS;/override command/ if(lc)/Dose the marker have a length
count/ state=GETLC0;
else state=LOOKING;
break;
case(GETLC0): loadlc0;/Load the top lengthcount byte/ state=GETLC1;
remove;
break;
case(GETLC1)
loadlc1;
remove;
state=DECLC;
break;
case(DECLC): lcnt=lcnt-2 state=CHECKLC;
break;
case(CHECKLC): if(lcnt=0)
state=LOOKING;/No more to do/ else if(lcnt<0)
```

state=LOOKING;/generate Illegal Length Error/ else state=COUNT;
break;
case(COUNT): decrement length count until1if(lc<=1)
state=LOOKING;
}
B.1.2.4 ″Input shifter (scinshft.sch, scinshm.M)″
The basic operation of this block is completely easy. This block takes out a data byte from an input circuit, loads it to a shift register, and shifts it. However, if it receives the BYPASS (flash plate of other SRs is carried out) command which processes the transition from/to a bypass mode also according to the command from an input decoder, the byte who unites is not loaded to a shift register. Instead, ″Labiche (rubbish)″ (tag =1) is shifted, and it forces so that the data currently held at other shift registers may be made to output. Then, a block waits for the signal which shows that this ″Labiche″ appeared in the token generator and ″by which the flash plate was carried out.″ An input byte is directly seen off in a token generator next.
[1149]B.1.2.5 ″Start code detector (scdetect.sch, scdetm.M)″
This block contains two programmable shift registers in 16 or 24-bit start code detection logic, and effective ″contents″ detection logic. Although an MPEG start code requires perfect 24 bits, H.261 requires only 16 bits.
[1150]Although it has a tag which directs whether the 2nd SR has an effective bit in data SR although the 1st SR is an object for data and there is not a gap or damping in SR (in meaning of a 2 line interface) in this invention, While the flash plate of the bit which they include is carried out, it may be invalidity (Labiche). If a start code is detected, it will be set up in order that the bit of a tag shift register may repeal the contents of the detector SR.
[1151]A start code must have been detected unless all the contents of the SR become effective. A flag indication of the non-byte alignment start code may be detected and given. If a start code is detected, a flag indication of the non-aligning start code must have been clearly given until the overlapping start code has been investigated. in order to achieve this function, the ″value″ of the detected start code (a byte follows it) lets scinshift and scdetect pass -- the right -- and a scoshift shift is carried out. If scoshift is reached without detecting another start code, the overlapping start code would be removed and a flag indication of it will be given as an effective start code.
[1152]B.1.2.6 ″Output shifter (scoshift.sch, scoshm.M)″
The basic operation of an output shifter is taking out serial data (and tag) from scdetect, filling it in a 15-bit word, and outputting. :B.1.2.6.1 which is as follows [functions / other] ″Data padding″
Although an output comprises 15 bit word, the input can consist of arbitrary numbers of bits. Therefore, in order to carry out a flash plate, and to carry out the last word by

15 bits, it is necessary to add a bit. These addition bit must be called padding, and must be recognized and removed by the Huffman block. In order that "zero" may be inserted after the data bit of the :last defined as follows and padding may complete 15 bit word, sufficient "thing" follows it.
[1153]Data word including padding is outputted by a low extension bit, in order to direct that it is in the end of a data token.
[1154]B.1.2.6.2 "Generating of a flash plate"
According to this invention, generating of "flash plate" operation detects the time of the flash plate of all the SRs being carried out, and includes signing an input shifter in it. If "Labiche" inserted by the input shifter reaches the end of an output shifter and an output shifter completes the padding, the signal "by which the flash plate was carried out" will be emitted. To the front where it is safe that an input shifter goes into a bypass mode, this signal "by which the flash plate was carried out" must pass a token generator.
[1155]B.1.2.6.3 "Flag display of an effective start code"
If it directs that scdetect discovered the start code, padding will be carried out and the present data will be outputted. Start code value (the next byte) is shifted through a detector, and removes an overlapping start code. If an output shifter is reached without detecting another start code "value", it does not pile up, but the value will attach the flag v not t (ValueNotToken) which shows that it is start code value, and will be sent. However, an overlapping error will be emitted, if the output shifter is waiting for the value while another start code is detected (scdetect). In this case, the 1st value is thrown away and a system waits for the 2nd value. This value can also be piled up, and the same procedure is made to be repeated until a non-overlapping start code is found in this way.
[1156]B.1.2.6.4 "Order after a start code (tyding up)"
If data (he is not Labiche) begins to arrive after detecting and outputting a suitable start code, new data headers will be made.
[1157]B.1.2.7 "Data stream generator (sctokrec.sch, sctokrem.M)"
Other one has a 2 line interface input from scoshift for the stuffed data and a start code from scinshift for the token by which one was bypassed as for the data stream generator. The change between two sauce is allowed only after the present token (from one of sauce) completes (low extension bit which arrived).
[1158]B.1.2.8 "Start value for starting number conversion (scdromhw.sch, schrom.M)"
The process of changing start value into a token is performed in two steps. This block mainly deals with the coding standard dependence problem of decreasing the position code of the odd number of 520 to the independent index of the coding standard of 16. As stated in early stages, start (value of JPEG is included) value is discriminated from other data of all the by a flag (value not token). v If not t is high, this block will change 4 or 8-bit value into 4-bit start number which is independently from a standard

according to a coding standard, and will indicate the unrecognized start code by a flag. The number of starts is as follows. : [1159]
[Table 171]
[1160]
[Table 172]
[1161]
[Table 173]
B.1.2.9 "The number of starts to token conversion (sconvert.sch, sconverm.M)"
In the 2nd step of conversion, the above-mentioned number of starts (or index) is changed into a token. This block processes the token extension and search mode which throw away extension and user data appropriately further.
[1162]Search mode is a means to go into a data stream on a random point. :0 which can set search mode as one of the eight value : The usual operation - The following start code is found.
[1163]1/2: A system level search is not carried out about a space decoder.
[1164]3 : the non-zero search mode which looks for the start code of the 4 which looks for the thing beyond a sequence next is a desired start code (or in syntax.). : 5 which looks for the thing more than a group : 6 which looks for the thing more than a picture : 7 which looks for the thing more than a slice : more -- it is high -- data is thrown away until it is detected.
[1165]This block adds token extension to PICTURE and a SLICE start token further.
[1166]- A picture start is extended by PICTURE NUMBER and four bit counts of a picture.
[1167]- A slice start is extended by svp (slice vertical position). This is the "value" of the start code minus 1 (MPEG, H.261) and the minus OXDO (JPEG).
[1168]B.1.2.10 "Data stream formatting (scinsert.sch, scinserx.M)"
In this invention, data stream formatting is related with a picture and a flash plate, a coding standard, conditional insertion of the token of a sequence start, and generating of a STOP AFTER PICTURE event. :switch which the function is best simplified in software and can be explained (input data)
case(FLUSH)
1.if(in picture)
output=PICTURE END 2.output=FLUSH 3.if(in picture & stop after picture)
sap error=HIGH in picture=FALSE;
4.in picture=FALSE;
break case(SEQUENCE START)
1.if(in picture)
output=PICTURE END 2.if(in picture & stop after picture)
2a. output=FLUSH 2b.sap. error=HIGH in picture=FALSE 3.output=CODING STANDARD 4.output=standard 5.output=SEQUENCE START 6.in-picture=FALSE;

break case(SEQUENCE END)
case(GROUP START): 1.if(in picture)
output=PICTURE END 2.if(in picture & stop after picture)
2a.output=FLUSH 2b.sap error=HIGH in picture=FALSE 3.output=SEQUENCE END or GROUP START 4.in-picture=FALSE;
break case(PICTURE END)
1.output=PICTURE END 2.if(stop after picture)
2a.output=FLUSH 2b.sap error=HIGH 3.in-picture=FALSE break case(PICTURE START)
1.if(in picture)
output=PICTURE END 2.if(in picture & stop after picture)
2a.output=FLUSH 2b.sap error=HIGH 3.if(insert sequence start)
3a. output=CODING START. 3b. output=standard3c.output=SEQUENCE START insert sequence start=FALSE 4.output=PICTURE START in picture=TRUE. break default:Just pass it through section B.2 ″The Huffman decoder and parser (parser)″
B.2.1 ″Introduction″
This section explains the Huffman decoder and parser circuit by this invention.
[1169]Drawing 127 shows the block diagram of a high level of the Huffman decoder and a parser. For plainness, many signals and buses are omitted in the main line figure, and there are some places to which especially data is sent back (within the illustrated big loop).
[1170]Intrinsically, the Huffman decoder and parser of this invention comprise many exclusive processing blocks controlled by a programmable State machine (shown along with the pars basilaris ossis occipitalis of a diagram).
[1171]Data is received from a coding data buffer by ″yne shift″ block. They are coding data which :data token which has the information on two types which will encounter intrinsically at this time owns, and the starred code already replaced with each token by the start code detector. Although other tokens may be encountered, all (except for a data token) the token is processed by the same method. A token (start code) is processed as a case of being special as much (setting to H.261, JPEG, or MPEG) data is coded after this.
[1172]In this invention, all the data which a data token owns is transmitted to the Huffman decoder with an in-series gestalt (bit by bit). Of course, it is coded by fixed length although this data includes many fields by which Huffman encoding is not carried out. Nevertheless, this data is sent to the Huffman decoder in series. In the case of Huffman encoding data, the Huffman decoder performs only the 1st stage of decoding with which a actual Huffman code is replaced with an index number. When N district Huffman code is shown in the special code table decoded, the range of this the ″Huffman index″ is zero to N-1. The Huffman decoder is sent without ″no op″ and processing get it blocked, have ″no operation″ mode, and according [it] to the

Huffman decoder to the following stage in accordance with data or token information.
[1173]A data unit index part is a comparatively easy circuit block which performs table-look-up operation. It takes out the name from the 2nd stage of a Huffman decoding process, and is changed into actual decipherment data by a table look-up with an easy index number obtained in the Huffman decoder there. A data unit index part cooperates with the Huffman decoder, and acts as one Logical unit.
[1174]ALU is the following block, and it is provided in order to add other conversion to decoded data. The data unit index part is suitable for comparatively arbitrary mapping, and arithmetic can use ALU for a more suitable place. ALU possesses the register file which can be operated in order to carry out various portions of decoding Argo RISUMU. The register which holds vector prediction and DC prediction especially is contained in the block of ****. ALU is based on the surroundings of a simple adder with operand selection logic. It includes the sign extension type dedicated communication circuit for operations. Although shift operation is easy to perform, this is carried out with a serial form and, probably, barrel shifter does not exist.
[1175]The token formatting part by this invention is the block of the last in a video parser.
Work eventually included in the token which can send decoded data to the remaining decoders is done.
At this time, there are many tokens which will be used by this specific decoder for pictures.
[1176]The parser State machine which is 8 bit width and is used with the 2-wire system interface does work which prepares the operation of other blocks. Intrinsically, it is the very simple State machine and makes the very wide range ″macrocode″ control word sent to other blocks. As for Drawing 127, an instruction word passes the data side from a block to a block. This is the actual condition and it is important to understand that transmission during a different block is controlled by a 2-wire system interface.
[1177]In this invention, a 2-wire system interface is between each block in a video parser. The Huffman decoder operates [data / a serial and] with a control token again, and a yne shifter inputs 1 bit of data at a time at once. Therefore, the operation in the two modes occurs. If data is inputted into the Huffman decoder via a data token, the data will pass 1 bit of shifters at a time at once. A 2-wire system interface is between a yne shifter and the Huffman decoder. However, other tokens are not shifted with a 1-bit gestalt at once (serially), but are rather shifted in the header of a token. If a data token is inputted, the header containing address information will be deleted and the next data of the address will be shifted at 1 bit at once. When it is not a data token, all the tokens, a header, and all the things are altogether displayed on the Huffman decoder at once.
[1178]In this invention, it is important for the 2-wire system interface for video

parsers to understand that it is exceptional at a point with two effective lines. One line is effective in a serial type, and the line of another side is effective in a token type. Both the line token may be unable to conclude simultaneously. Since both cannot be concluded even if there are two effective lines if line of one of the two is concluded or effective data does not exist, it should recognize that there is only one acceptance wire in other directions. However, this does not pose a problem. It can be known any the Huffman decoder shall desire between serial data or token information if needed that it should be accomplished next based on the present syntax. Therefore, an effective accept signal is set up according to it, and Accept is sent to a yne shifter from the Huffman decoder. If suitable data or token exists, a yne shifter will send an effective signal. For example, typical instructions can decode a Huffman code, it can be changed in a data unit index part, that result can be corrected in ALU, and this result is formed in a token word. One microcode instruction word including all the information for carrying this out is made. A command is directly sent to the Huffman decoder, and the Huffman decoder requires one data bit at a time from a ″yne shift″ block until it finishes decoding a perfect sign. A control token is inputted in parallel. Once this occurs, the decoded index value will be sent to a data unit index part with the original micro code word. In order that the Huffman decoder may perform this operation, some cycles are needed, and the number of cycles is actually determined by the data decoded. A data unit index part maps this value using the table pinpointed in a microcode instruction word next. This value is again sent to the following block and ALU with the original micro code word. operation with suitable ALU -- completing (the number of cycles may be data subordination too) -- ALU sends the suitable data for a token formatting block with the micro code word which controls the method of forming a token word.

[1179]ALU has the ″condition code″ returned to many status wires or parser State machines. The State machine enables it for this to perform condition jump instructions. In fact, one of the conditions in which all the instructions are condition jump instructions and which are made as for :selection is wired by value ″False″. ″No-jump″ instructions can be constituted by choosing this condition.

[1180]According to this invention, a token formatting part has the fixed field containing the data field from two input:ALUs and/, or a parser State machine. Besides, in addition, what bit is taken from one sauce and there are instructions which tell a token formatting part whether it should fill on the whole in the remaining bits of other sauce for 8 bits. For example, HORIZONTAL SIZE has eight bit fields which are the eternal addresses which specify it as a HORIZONTAL SIZE token. In this case, there is no data which is obtained from the fixed field 8 bits and obtained from ALU. However, if it is a data token, 6 bits will be easily obtained from the fixed field, and 2 bits of the lower one show the color component from ALU. Therefore, a token formatting part acquires this information, and in order to use it by the remaining

systems, it places it into a token. In the above-mentioned example, the number of bits from each sauce is only for for the purpose of explanation, and if it is a person skilled in the art, it should note recognizing that the number of bits from which sauce can also be changed.
[1181]ALU includes the counter bank used in order to count through the structure of a picture. The dimension of a picture is programmed by the register which unites with the counter which appears in a "micro programmer" as a part of register bank. Some condition codes are outputted from this counter bank, and it makes possible the condition jump based on a "picture start", "a macro block start", etc.
[1182]What should be careful of is that a parser State machine is also called a "demulti PUREKKUSU State machine." The word in two ways is used in this document.
[1183]In input shifter this invention, an input shifter is a very simple circuit which comprises two pipeline stage data paths "hfidp" and control Zcells"hfi."
[1184]Token decoding happens in the 1st pipeline stage. Only a data token is recognized in this stage. 1 bit of data contained in a data token is shifted to the Huffman decoder at once. The 2nd pipeline stage is a shift register. In the last word of a data token, special coding occurs so that the arbitrary numbers of bits can be transmitted through a coding data buffer. In last data word, all the possible patterns are described below.
[1185]
[Table 174]
The bit pattern "all the numbers following 0" is looked for as a data bit is shifted to the one left at a time in a shift register (padding). As for this, the remaining bits in a shift register are not effective, and being thrown away is shown. It notes generating this operation only in the last word of a data token.
[1186]As mentioned above, other tokens of all the are in parallel with the Huffman decoder, and are sent. A shift is not generated although they are loaded also to the 2nd pipeline stage. It notes that data headers are thrown away and are not sent to Huffman at all. Two "effective" wires (out valid and serial valid) are provided. Only one is concluded at predetermined time and it directs which type of data is displayed on the moment.
[1187]B.2.2 "Huffman decoder"
The Huffman decoder has many operation modes. The most remarkable thing is that it decodes a Huffman code and changes them to the Huffman index number. Besides, in addition, the Huffman decoder can decode the fixed length code of the length determined by an instruction word (in bit). The Huffman decoder can receive the token from yne shift blocks further.
[1188]The Huffman decoder is the very small State machine. This is used when decoding the information on a block level. This is because it starts determining a

parser State machine too much for a long time (before it makes a decision about data and takes out a new command, it is because it must wait for data to flow through a data unit index part and ALU). When this State machine is used, the Huffman decoder itself emits a command to a data unit index part and ALU. Since the Huffman decoder State machine cannot control all the microcode instruction bits, the command of a total range cannot be emitted to other blocks.

[1189]B.2.2.1 "Theory of operation"

When decoding a Huffman code, the Huffman decoder of this invention uses an operation procedure, in order to decode input code to the Huffman index number. This number is between zero (as opposed to code table with N entries) to N-1. One bit is received at a time from an input shifter.

[1190]Many tables are required in order to control the operation of a machine. It is specified whether for each possible number of bits, these have a code of the length however in a code (1-16 bits). Since a typically general Huffman code is specified, this information is not enough so that it may be expected. However, in MPEG, H.261, and JPEG, a Huffman code is chosen so that a Huffman code table can be specified only for this information. : which has only one exception in an unfortunate thing at this -- it is a Tcoefficient table from H.261 used also in MPEG. This needs the additional table explained at other places (the exception was intentionally introduced into H.261, in order to avoid a start code emulation).

[1191]It is important for the table which this Huffman decoder uses to recognize what is transmitted to JPEG, and the correctly same thing. While possibly this required creation of the internal table from what other designs of the Huffman decoder were delivered, it makes it possible to use these tables directly. Probably, this required additional memory storage and additional processing for conversion. Being able to explain the table of MPEG and H.261 similarly (there is an above-mentioned exception), a multi-standard decoder becomes practical.

[1192]:int total=0 the following "C" fragmentation explains a decoding process to be;

```
int s=0;
int bit=0;
unsigned long code=0;
int index=0;
while(index>=total)
{if(bit>=max bits)
fall ("huff decode: ran off end of huff table**n");
code=(code<<1)Inext bit0;
index=code-s+total;
total+=codes per bit[bit];
s=(s+codes per bit[bit])<<1;
bit++;
```

} Generally, specific middle value has an advantage in the fact that it is calculable in a clock phase, before they are needed, but direct mapping of the process is carried out into silicon mounting.
[1193]It is :EQ1. $total_{n+1}=total_n+cpb_n$EQ2.$'s_{n+1}=2\ ('s_n+cpb_n)$ that code fragmentation shows.
Bad [EQ3. $code_{n+1}=2code_n+bit_n$EQ4. $index_{n+1}=2code_n+bit_n+total_n-'s_n$ fate] in hardware. It turned out that it is easier to use the corrected formula of a series used instead of a "finishing [a shift]" variable being a variable "S." In this case, in :, however hardware, it turned out that it is easier to use the corrected formula of a series used instead of a "finishing [a shift]" variable being a variable "S." in this case, : -- EQ5. $shifted_{n+1}=2shifted_n+cpb_n$ -- : which the following thing understands, if EQ6. $i_n=2shifted_n$, therefore this are again transposed to the formula 4, :EQ7 from which the following formula is made -- calculating the value which .$index_{n+1}=2(code_n-shifted_n)+total_n+bit_n$ "index" follows -- in addition, it is required to know when calculation will be completed. : from the "C" code fragmentation -- EQ -- it turns out that it completes at the time of 8. $index_{n+1}<total_{n+1}$.
[1194]When it replaces from the formula 7 and the formula 1, it turns out that it completes at the time of EQ9. $2(code_n-shifted_n)+bit_n-cpb_n<0$.
[1195]In hardware mounting of this invention, the common term in EQ.7 and EQ.9 and $(code_n-shifted_n)$ are calculated before 1 phase when the remainder of these formulas is evaluated at and the end product and calculation take out the information "it completed."
[1196]It is a warning word One In various "C" codes especially an edited [action] code Huffman decoder, and a sm4code project, although "C" fragmentation is used almost directly, a variable "S" is called "finishing [a shift]" in practice. Thus, there are two different variables called "finishing [a shift]." It is one of the "C" codes, and other one in hardware mounting. These two variables change with two factors.
[1197]B.2.2.1.1 "Reversal of a data bit"
There is one more information required in order to decode a Huffman code correctly. This is the polarity of coding data. It turns out that H.261 and JPEG use opposite regulation. The marker byte of JPEG is produced from the fact of being 1 bit, to the start code of H.261 of this being a zero bit.
[1198]In order to process both regulations, in order that coding data may decode the Huffman code of an H.261 style, when read into the Huffman decoder, it is required to reverse a coding data bit. This is performed by the clear method of using a monopolistic OR gate. Since data is not reversed when decoding a fixed length code, it notes that reversal is performed only for a Huffman code.
[1199]MPEG uses the mixture of two regulations. H.261 regulation is used in the aspect of affairs succeeded from H.261. JPEG regulation is used in the aspect of affairs succeeded from JPEG (DC intra decoding of a coefficient).

[1200]B.2.2.1.2 "Conversion factor table"
When using the conversion factor table in H.261 and MPEG, there are some strange rules. An MPEG table is a superset of an H.261 table the 1st. There is no distinction pulled between two standards, and this as used in hardware mounting of this invention means that the H.261 stream containing the code from the extension of a table (that is, the MPEG code) will be decoded by the "right" method. Of course, other aspects of affairs of a compression standard are destroyed similarly. For example, these extension codes will produce the start code emulation in H.261.
[1201]A conversion factor table has a strange rule which means what cannot be explained by the usual method of using a code per bit table in the 2nd. This strange rule is generated in code of 6 bit length. These code words are systematically replaced by a shift code word. In an encoder, a right result is obtained by the usual method by the 1st encoding. Next, 6 bits of the beginning are replaced by easy table-look-up operation at another 6 bits for all the codes of the length beyond 6 bits or it. In the decoder by this invention, a decoding process is interrupted just before the 6th bit is decoded, a code word is replaced using a table look-up, and decoding is continued.
[1202]In this case, there is only a possible six bit code of 10, therefore a required look-up table is dramatically small. Operation is helped by the fact that top 2 bits of a code are not changed by operation. It becomes unnecessary as a result, to use a true look-up table. Instead, a small gate group is wired and suitable conversion is performed. The module which performs this is called "hftcfrng". By that with which each code from a possible code set is replaced in another code from that set (the old code is omitted or new chord is not introduced), this type of code substitution is specified as a "ring" in the text.
[1203]Mounting peculiar for the coefficient of the beginning within a block is performed. In this case, a table is corrected so that the sign which that a block end code occurs most generally generates impossible therefore can use the code interpreted as a block end if that is not right. Thereby, 1 bit can be saved. With the composition for decodings by this invention, this is accommodated easily. If it says simply and an "index" will be value zero, it will be considered that decoding was ended for the 1st bit of the 1st coefficient. After decoding only 1 bit, it is only required for "indexes" to have only value and zero with two possibilities, and 1, and to test 1 bit.
[1204]B.2.2.1.3 "Register and size of an adder"
The Huffman decoder of this invention can process the Huffman code up to 16 bits. However, a decoding machine is 8 bit width. Since it turns out that the Huffman index numerical value decoded is 255 at the maximum, this is possible. In fact, this can be generated only in extended JPEG and the limit is low (however, since it is larger than 128, 7 bits is insufficient) a little in the present application.
[1205]It turns out to all the lawful Huffman codes that not only the last value of an

″index″ but all the mean values are in the range of 0-255. However, to the attempt which is going to decode an illegal code and the code which is got blocked (probably for a data error) and is not in the present code table, an index value may exceed 255. Since we are using an 8-bit machine, it is a decoding end and the last value of an ″index″ does not exceed 255. It is because the important bit is thrown away rather than telling us that the error occurred. For this reason, when the index value under decoding exceeds 255 (carry from the adder which is got blocked and forms an index), an error occurs and decoding is abandoned always.
[1206]12 bits of a ″code″ are protected. This is not required, in order to come out enough with three bit registers and to decode a certain Huffman code. The high order bit of these is required because of the fixed length code which can be read to 12 bits.
[1207]B.2.2.1.4 ″Operation for fixed length codes″
It is forced so that ″code for every bit″ value may become zero for a fixed length code. ″Finishing [a shift] of this″ as the ″whole″ continues being zero through operation, therefore the ″index″ is the same as a code. In fact, as for adders, only 8-bit value is made for an ″index.″ For this reason, when decoding a fixed length code, the high order bit of an output word is taken directly from a ″code″ register. When decoding a Huffman code, it is forced these high order bits so that it may become zero.
[1208]The fact that sufficient bit was read from the input is calculated by a clear method. A comparator compares the desired number of bits with a ″bit″ counter.
[1209]B.2.2.2 ″Decoding coefficient data″
Generally the parser State machine by this invention is considerably used only for decoding of a high level. Decoding of a low is not directly processed by this State machine very much in an every 8 bits data block. A parser State machine takes out a command to the Huffman decoder of a gestalt ″which decodes a block.″ Both the Huffman decoder, a data unit index part, and ALU operate under control of the exclusive (it is in the Huffman decoder fundamentally) State machine. This arrangement makes possible highly efficient decoding of entropy coding coefficient data. Other feeding back passes operated in this operation mode occur. For example, in JPEG decoding by which VLC is decoded in order to provide SIZE and RUN information, It is directed to the Huffman decoder whether SIZE information is directly returned to the Huffman decoder from the output of a data unit index part, and should read many FLC bits however. Besides, in addition, some accelerators are mounted. For example, all the VLC value which produces SIZE of zero is clearly captured using the same example by considering the Huffman index value in front of a data index stage. Before SIZE value with the actual Huffman decoder understands this in the case of non-zero SIZE value, it means that it can progress in order to read one FLC bit. Since this laps with one clock cycle required in order that read-out of the first FLC bit may carry out a table look-up in a data unit index part, it means that any clock cycles are not wasted.

[1210]B.2.2.2.1 ″MPEG and AC coefficient data of H.261″
Drawing 133 shows the method by which an AC coefficient is decoded in MPEG and H.261. The flow chart which shows the operation of the Huffman decoder in detail is shown in Drawing 128.

[1211]A process starts by reading the VLC code. In the usual event, direct mapping of the Huffman index is carried out to the value showing 6 bits of absolute values of RUN and a coefficient. Then, 1 bit of FLC is read and the sign of a coefficient is taken out. ALU assembles the absolute value of a coefficient by this sign binary digit, and provides the last value of a coefficient.

[1212]Since the data format in this time is sign magnitude, there is no almost difficult thing in this operation. While RUN value is sent to a 6-bit auxiliary bus, coefficient value (level) is sent to the usual data bus.

[1213]There are two cases of being special and these are captured by considering the index value decoded before data index operation. These are a block end (EOB) and escape code-ized data. In the case of EOB, the fact that this occurred is sent through a data unit index part and an ALU block so that the data token by which the token formatting part was opened can be closed correctly.

[1214]Escape code-ized data is more complicated. The first 6-bit RUN is read, and these are directly sent through a data unit index part, and are memorized by ALU. Then, 1 bit of FLC is read. This is the most important bit among 8 bits of the escape explained in MPEG and H.261, and gives the sign of a level. Since it is necessary to send a command which is different in ALU for negative value pair positive value, the sign is clearly read in this mounting. ALU enables it for this to change two complement values in a bit stream into sign magnitude. In any case, 7 bits of the remaining FLC are read after that. 8 bits must be read if this is value zero.

[1215]In this invention, in order that the internal State machine of the Huffman decoder may also control the data unit index part for own control, ALU, and a token formatting part, it is responsible for generating a command. As shown in Drawing 133, instructions of the Huffman decoder are sent from one of the instructions memorized by the register received from three sauce; parser State machines, the Huffman State machine, or the parser State machine before. Intrinsically, the primitive instructions from a parser (control is made to take over to Huffman State machine, and coefficient is made to read) State machine are used whenever [for which it is got blocked and new VLC is needed] it is held at a register. Other the instructions of all the for decoding are supplied with the Huffman State machine.

[1216]B.2.2.2.2 ″DC coefficient data of MPEG″
This is processed like the DC coefficient data of JPEG. It is the responsibility for a control microprocessor for the same (loading is possible) table to be used and to guarantee that the contents are right. The only realistic difference with an MPEG standard is that a predictor is set as zero and the amendment is performed in an

inverse quantization device (to the case and the appearance of JPEG).
[1217]B.2.2.2.3 ″JPEG coefficient data″
Drawing 129 is a block diagram showing the hardware for decoding the AC coefficient of JPEG according to this invention. Since the process for DC coefficients simplifies a JPEG process intrinsically, a diagram acts for both AC and a DC coefficient. The only thing actually added to the diagram of the front for the AC coefficients of MPEG is being able to use it as a part of Huffman decoder command in order to return the ″SSSS″ field and to specify the FLC number of bits read. The remaining commands are supplied with the Huffman State machine.
[1218]Drawing 130 is drawing the flow chart for AC and Huffman decodings of a DC coefficient.
[1219]Although the process for AC coefficients is taken up first, a process is started by reading VLC using a suitable table (there are two AC tables). The Huffman index is changed into RUN and the SIZE value of data unit index circles next. Although two value is captured in the Huffman index stage, these are EOB and an object for ZRL. These are two value by which any FLC bits are not read uniquely. When a decoding index is not a thing of these two value, it waits to determine how much bit while the Huffman decoder reads 1 bit of FLC promptly, a data unit index part completes lookup operation, and is actually required for it. In the case of EOB, the further processing that the Huffman State machine in the Huffman decoder carries out is not required, and another command is read from a parser State machine. In ZRL, any FLC bits are not required, but a block is not completed. In this case, the Huffman decoder starts promptly the decipherment of further (using the same table as before) VLC.
[1220]Although the index value which unites to ZRL and EOB is detected, there is a special problem. This is because the Huffman table is downloadable (H.261, MPEG, and difference). For [each] AC table of two JPEG, two (as for one, one is for EOB for ZRL) registers are provided. These are loaded when a table downloads. They hold the index value which unites with a suitable sign.
[1221]ALU must change the FLC code of a SIZE bit into suitable sign magnitude value. These are loaded when a table downloads. They hold the index value which unites with a suitable sign. ALU must change the FLC code of a SIZE bit into suitable sign magnitude value. This is performed by carrying out sign extension of the value with the first wrong sign. If a sign binary digit is newly set up, the remaining bits will be made reverse next (one's complement).
[1222]Since there is no equivalent of the ZRL field in the case of a DC coefficient, the decision making in a Huffman decoding stage is easier a little. The only sign with which the FLC bit of zero is read shows DC difference of zero. This is again captured in the Huffman index stage, and a register is provided in order to hold this index on each DC tables of JPEG (it is downloadable).
[1223]By holding the copy of the last (known as prediction) DC coefficient value, ALU

of this invention does work of forming a final decoded DC coefficient. Every one a total of four predictors are required because of [each] four activity color components. If DC difference is decoded, ALU will form the value which added the suitable predictor and was decoded. This is too memorized as a predictor for the next DC difference of the color component. Since a DC coefficient is signed (for DC offset), it needs the conversion for sign magnitude from two complements. Then, the value is outputted with RUN of zero. In fact, the instructions "carry out a part of this last stage" are not supplied with the Huffman State machine. They are only performed with a parser State machine.
[1224]In the same method of receiving an AC coefficient, ALU must generate DC difference from the SIZE bit of FLC first. However, it is called for that two complement values are added to a predictor in this case. This is formed by carrying out sign extension with the first wrong sign as mentioned above. If the result is negative, 1 must be added in order to form a right value. Of course, this may be added simultaneously with a predictor by stuffing carry into an adder.
[1225]B.2.2.3 "Error processing"
About error processing, it is worthy of explaining. For the error detected, it separates from : and the table end which has four sauce in practice.
[1226]- It is serial when a token is expected.
[1227]- It is a token when a serial is expected.
[1228]- Many coefficients are in a block too much.
[1229]The first thing is generated in two situations among these. Since the longest lawful Huffman code is 16 bits when a bit counter amounts to 16 (lawful value is 0-15), an error occurs. When the mean value of an "index" exceeds 255, as it explained in the section B.2.2.1.3, an error occurs.
[1230]The 2nd thing is generated when encountering serial data that a token is expected. The 3rd thing is generated when an opposite state arises.
[1231]It generates, when the error of the last type has too much many coefficients in a block. This is actually detected in a data unit index part.
[1232]If either of these states occurs, an error will attract attention in the Huffman error register, and a parser State machine will be interrupted below it. It is the responsibility for a parser State machine to process an error and to emit a command required for recovery.
[1233]Huffman cooperates with a parser State machine at the time of interruption for guaranteeing right operation. If the Huffman decoder interrupts a parser State machine, it can wait to accept a new command in the output of a parser State machine. The Huffman decoder will not accept this command for two complete cycles, after interrupting a parser State machine. Thereby, the parser State machine can remove the command which is there (it should not perform now), and can replace it with a suitable thing. After these two cycles, the Huffman decoder starts the

operation of ****** again, and if an effective command is there, it will accept the command. Otherwise, nothing will be done until a command with an effective parser State machine is shown.
[1234]If the ″Huffman error″ event bit will be set up if either of these errors occurs, and a mask bit is set up, it stops and the control microprocessor will be interrupted below the block by the usual method.
[1235]It may not be an error which looks like an error in a certain situation actually. The most important situation that such a thing generates is a time of reading the macro block address. In the syntax of MPEG, H.261, and JPEG, it is lawful that a token occurs instead of the macro block address expected. if such a thing occurs by the lawful method, zero load to the Huffman error register -- having (it means being without error) -- a parser State machine is still interrupted. The code of a parser State machine recognizes the situation of this no error, and answers according to it. In this case, the ″Huffman error″ event bit is not set up and the block will not suspend processing.
[1236]Various situations must be processed. First, a token is generated promptly and a serial bit does not move forward. In this case, although ″the token error in case a serial is expected″ probably occurred instead, it generates by a method which the no error error mentioned above.
[1237]The serial bit of 2-3 is [2nd] before a token. In this case, it succeeds in determination. When all the bits before a token have the value 1, (my wanting you to remember that there is a zero bit in a coding data file in H.261 and MPEG, since coded data's is reversed), and a no error occur. however -- if one of them is zero -- a stuffing bit with effective them -- not but, an error occurs in this way and a ″token in case serial is expected″ error occurs.
[1238]Many bits are [3rd] before a token. In this case, it succeeds in the same determination. If all the 16 bits are 1, they will be processed as padding bits and a noerror error will generate them. When either of them is zero, the error ″which separates from the Huffman table″ occurs.
[1239]Somewhere else which is generated without expecting a token is JPEG. When processing either the Huffman table or a quantizer table, the table of any numbers can be generated in the same marker segment. It cannot be known how many tables the Huffman decoder has. For this reason, after each table is completed, the Huffman decoder reads 4 bits of another FLC, and assumes it to be new table numbers. However, if a new marker segment starts, the token will be encountered instead of being 4-bit FLC. This requirement was not predicted, therefore the Ignore Errors command bit was added.
[1240]B.2.2.4 ″Huffman command″
In order to control the Huffman decoder blocks and those definitions, the bit which a parser State machine uses is shown here. A data unit index part command bit is also

contained on this table. From a micro programmer's viewpoint, the Huffman decoder and a data unit index part act as one condensed logical block.
[1241]
[Table 175]
B.2.2.4.1 "FLC read-out"
In this mode, all of Ignore Errors, Download, Alutab, Token, First Coeff, and Special and VLC are zero. Bypass is set up so that data index conversion may not occur.
[1242]It is directed how the binary number of a table [3:0] should read many bits. The numbers 0-12 are lawful. Value zero actually (probably, it was expected like) read a zero bit, therefore these instructions are Huffman decoder NOP instructions. The value 13, 14, and 15 does not operate, but when the value 15 has the Huffman State machine under control and use of "SSSS" is displayed as the number of bits of FLC which should be read, it is used.
[1243]B.2.2.4.2 "VLC read-out"
In this mode, Ignore Errors, Download, Alutab, Token, First Coeff, and and Special are zero, and VLC is 1. Usually, Bypass is set up so that data index conversion may occur.
[1244]In this mode, all of Token, First Coefficient, and Special are zero, and VLC is 1.
[1245]The binary number of a table [3:0] directs the table which should be used as shown below.
[1246]
[Table 176]
In the case of the table (that is, JPEG table) held at RAM, the bit 1 is not used but it notes that table selection occurs twice. When a non-baseline JPEG decoder is made, there are four DC tables and four AC tables, and a table [1] is needed.
[1247]Since a table will be correctly read as a table of an H.261 style if a table [3] is zero, input data is reversed when used. In the case of table [3:0] =0, a suitable Ring correction is also applied.
[1248]B.2.2.4.3 "NOP instructions"
As mentioned above, the operation which reads FLC of a zero bit is used as No Operation instructions. Any data (token or it is serial someday) is not read from an input port, but the Huffman decoder outputs the data value of zero with an instruction word.
[1249]B.2.2.4.4 The "1st coefficient of TCoefficient"
H.261 and the TCoefficient table of MPEG have a special non-Huffman code used for the 1st coefficient of a block. In order to decode TCoefficient in a block start, a First Coefficient bit may be set as table zero with VLC instructions. One of many the effects of a First Coefficient bit is making it possible to decode this code.
[1250]In the usual operation, it notes that it is exceptional to emit an "easy" command in order to read TCoefficient VLC. By usually setting up Special Bit, this is because control is handed to the Huffman decoder.

[1251]B.2.2.4.5 "Read-out of a token word"
In order to read a token word, the Token bit should be set as 1. Special and the First Coefficient bit should be zero. The VLC bit should also be set up when making a table [0] bit act correctly.
[1252]In this mode, in order that a bit table [1] and a table [2] may correct a token read-out act as follows, it is used. : [1253]
[Table 177]
If a table [0] and a table [1] are zero, it will be thought that existence of the serial data in front of a token is an error, and such a signal will be emitted.
[1254]When a table [1] is set up, all the serial data are thrown away until they encounter a token word. Existence of these serial data, therefore an error will not be generated.
[1255]Padding bits will be thrown away when a table [0] is set up. Of course, it is required to get to know the polarity of padding bits. This is determined by the table [3] by the completely same method as the case for read-out of VLC data. If a table [3] is zero, input data will be reversed first and one of "1" bits will be thrown away. When a table [3] is set as 1, input data is not reversed but "1" bit is thrown away. Since operation of reversing data according to a table [3] bit is conditional in a VLC bit, this bit must be set as 1. An error is reported when encountering the bit which is not padding bits (it is got blocked and is "1" bit in H.261 and MPEG).
[1256]In these instructions, it notes that only one token word is read. It is the responsibility for a demultiplexer for the state of an extension bit to be disregarded, to test this bit, and to act according to it. The instructions which read a multiple word are also provided. - Refer to the section about instructions specially.
[1257]B.2.2.4.6 "An ALU register specifies a table."
If an Alutab bit is set up, the table numbers actually used using the register in an ALU register file can be determined. The table numbers supplied by a command with a VLC bit determine which ALU register is used. : [1258]
[Table 178]
In the case of a fixed length code, in order to carry out decoding of the vector, the right number of bits is read. r If size is zero, NOP instructions will arise.
[1259]In the case of a Huffman code, the table numbers to generate are the tables [3] set as 1, and the number produced as a result quotes one of the JPEG tables.
B.2.2.4.7 "Special instructions"
All the instructions (or operation mode) explained until now are taken into consideration as "simple" instructions. A suitable quantity of input data is read for each command received (even if it is any of serial data or token data), and the data produced as a result is outputted. One output will occur for every command correctly, if an error is not detected.
[1260]In this invention, it has the feature specially that one or more output words can

generate instructions for one command. The internal State machine of the Huffman decoder controls, and instructions will be issued in itself if needed until it determines that the instructions which the parser required were completed, in order to achieve this function.
[1261]Actual instructions of the beginning of the sequence which will be performed are taken out with the Special bit set as 1 in all the special instructions. This means that all the sequences must have the first peculiar instructions. The advantage of this attempt is being able to use the instructions with the beginning of a sequence actual without lookup operation demanded based on the command received from a parser.
[1262]The 1st thing of : and TCoefficient-JPEG DC-JPEG AC-Token with four special instructions recognized reads H.261, an MPEG conversion factor, etc. until block - and a sign are read. This command will read the whole block, when a block is the non-Intra block. In this case, the First Coefficient bit should be set up so that the 1st coefficient trick may be applied. If a block is the Intra block, DC term should already be read and the First Coefficient bit should be zero.
[1263]In the Intra block of H.261, DC term is read using instructions "simple" in order to read 8 bits of FLC value. In MPEG, special instructions of JPEG DC explained in the following are used. The JPEG DC command is used in order to read DC term of a JPEG (SSSS bit of FLC shown by VLC is included) style. Since a counter (the number of coefficients is counted) is reset in a data unit index part, a First Coefficient bit must be set up. The JPEG AC command is used in order to read the remainder of a block, after DC term until it encounters EOB or the 64th coefficient is read.
[1264]The Token command is used in order to read all the tokens. A token word is read until an extension bit becomes clear. It is a useful method of processing an unrecognized token.
[1265]B.2.2.4.8 "Download table"
In this invention, the Huffman decoder table is downloadable using a Download bit. The first step is nominating the table which should be downloaded. This is a Download bits set and a First Coeff bits set, and is performed by taking out the command for reading FLC. Although this is processed as NOP, therefore any bits are not read actually, table numbers are memorized by the register, and since it specifies which table is loaded in the next downloading, they are used.
[1266]
[Table 179]
As shown in the above-mentioned table, either AC or DC table can be loaded and it is determined whether tables [3] are whether it is a codes-per-bit (it can set to Huffman decoder itself) table, and a data index table loaded.
[1267]Once a table is nominated, data will be downloaded in the table by taking out the command for reading a required number of FLC (it is always 8 bits) with a Download bits set (and First Coeff bit zero). This is written in into the table where

coding data was nominated. An address counter is maintained, data is written in the current address, and then an address counter is increased. An address counter is reset by zero always, when a table is nominated.
[1268]When downloading a data index table, data and an address are supervised. While an address is the Huffman index number, it notes that the data loaded to the address is a sign decoded eventually. Since the register which holds the Huffman index number for a related sign is loaded automatically, this information is used. Therefore, when data with the value corresponding to ZRL is recognized in a JPEG AC table, the current address so that it may be directed by table numbers, It is written in the register CED H KEY ZRL INDEX0 or CED H KEY ZRL INDEX1.
[1269]Since the decoded data is written in a codes-per-bit table after 1 phase when it was decoded, data cannot be read from a table during this phase. Therefore, the instructions which try to read VLC taken out promptly after table download instructions will go wrong. In actual application, there is no reason which this **** sequence generates (when getting it blocked and performing JPEG). However, it is possible to build the simulation test which performs this.
[1270]B.2.2.5 The "Huffman State machine"
The Huffman State machine by this invention acts in order to provide the Huffman decoder command which is made internally in a certain case. The Huffman decoder may be provided with all the commands which may be made with an internal State machine by the demultiplexer.
[1271]The basic structure of the State machine is as follows. If a command is taken out to the Huffman decoder, it is memorized by a series of auxiliary latches, and a reuse can be carried out to behind. A command is executed by the Huffman decoder and analyzed by the Huffman State machine. If it is recognized as a command being a thing of the beginning of a publicly known instruction sequence and a SPECIAL bit is set up, the Huffman decoder State machine will succeed control of the Huffman decoder from a parser State machine. :1 parser State machine with which three sauce is in the instructions for Huffman decoders at this time: This choice is specially performed at the time of completion of instructions (for example, when EOB is decoded), and the following demultiplexer command is accepted.
[1272]2) Huffman State machine. The Huffman State machine can have arbitrary commands.
[1273]3) Primitive instructions issued by the parser State machine in order to start instructions.
[1274]In the case of (2), table numbers can be provided by the feedback from a data unit index part, and this will exchange the field in Huffman State machine ROM.
[1275]In the case of (1), in a certain case, table numbers are provided by the value obtained from an ALU (in for example, case of AC, DC table-numbers, and F-number) register file. These value is memorized by auxiliary command memory storage, and

table numbers are memorized when the reuse of the command is carried out to behind. Since the counter is advancing in order to usually quote the following block, it is not again recovered from ALU.
[1276]Since the choice of instructions of the next used is dependent on the data currently decoded, determination needs to be carried out in the direction of the last of a cycle. Therefore, all the possible instructions are prepared in parallel and a general structure opts for instructions with actual multiplexing in the second half of a cycle.
[1277]It adds to opting for the instructions which the Huffman decoder will use in the following cycle in each case, State machine ROM also notes opting for the instructions which will be attached to the present data, when the present data passes from a data unit index part and it to ALU. By the completely same method, these three instructions of all are prepared in parallel, and a choice is performed in the second half of a cycle.
[1278]There are three choices as mentioned above too also here for this portion of the instructions corresponding to three choices for the next Huffman decoder instructions.
[1279]1) Fixed instructions suitable to a block end.
[1280]2) Huffman State machine. The Huffman State machine can provide instructions arbitrary for a data unit index part.
[1281]3) Primitive instructions which a parser issues in order to start instructions.
[1282]B.2.2.5.1 "EOB comparator"
The output of an EOB comparator is forced so that the selection of constant instructions may be intrinsically displayed on a data unit index part, and it is made for the next Huffman instructions to be instructions of the next from a parser. The exact function of a comparator is controlled by the bit in Huffman State machine ROM.
[1283]Behind an EOB comparator, four registers holding the index of an EOB sign are shown in the JPEG table of AC and DC. In the case of DC table, of course, there are not block - and a sign, but there is a sign of zero size and it is made by DC zero difference. Since the zero bit of FLC is read completely like the case of an EOB sign as for this, they are completely processed in a similar manner.
[1284]In addition to three index value held at a register, the constant value 1 can also be used. This is an index number of the EOB sign in H.261 and MPEG.
[1285]B.2.2.5.2 "ZRL comparator"
In this invention, this is a general-purpose comparator. It makes either of the original instructions for being used by the Huffman State machine instructions or I to D choose.
[1286]There is four value behind a ZRL comparator. Two are in a register and they hold the index of the ZRL code in AC table. Other two value is constants, one is value zero, and other one is 12 (index of ESCAPE in MPEG and H.261). In the case of FLC, constant zero are used. The constant 12 is used always, when table numbers are

smaller than 8 (and VLC). One of the two registers is judged by the low order bit of table numbers, and when table numbers are larger than 7 (and VLC), it is used.
[1287]The bit in State machine ROM is provided in order that it may be provided in order to make a comparator possible, and another bit may reverse the operation. When the TOKEN bit in instructions is set up, a comparator output is disregarded and is replaced by the extension bit. This is continued to the end of a token.
[1288]B.2.2.5.3 ″Huffman State machine ROM″
The instruction field in the Huffman State machine is as follows, and is certain :nxtstate[4:0].
It is an address used in the following cycle. This address is correctable.
Correction of the State address of primary statect is enabled. If it is zero, the State machine address will not be corrected, otherwise, LSB of an address will be replaced at one value of the following two comparators. : [1289]
[Table 180]
Notes: If the next Huffman instructions are chosen as a ″rerun primitive command″ in any case, the State machine will be jumped for the location 0, 1, 2, or 3 as a thing suitable for a command.
eobct[1:0]
This controls the following selection of the Huffman instructions based on the following EOB comparators and an extn bit. : [1290]
[Table 181]
zrlct[1:0]
This controls the following selection of the Huffman instructions based on a ZRL comparator. If conditions suit, the State machine instructions will be taken, otherwise, rerun of the primitive instructions will be carried out. In any case, if eobctl+ conditions take demultiplexer instructions, this (eobctl+) has a right of priority as follows. : [1291]
[Table 182]
smtab[3:0]
In this invention, if the selected instructions are the State machine instructions, this is table numbers used by the Huffman decoder. However, if it is a ZRL comparator machine, it will be used by the zrltab [3:0] field, giving priority.
[1292]Probably, smtab [3:0] and zrltab [3:0] have the same value, when another table numbers do not need to be used according to whether ZRL conformity occurs. However, it notes that this may draw a strange simulation problem in Lsim. In the case of MPEG, there is no clear necessity of loading the register which directs the Huffman index number for ZRL (composition of only JPEG). However, these are still chosen, and when ZRL comparators are all that may be ″strangeness″ (therefore, it is not serious if which chosen), In spite of the fact that smtab [3:0] and zrltab [3:0] have the same value, a ZRL comparator will serve as ″strangeness″ and the following State will also become ″strangeness.″

zrltab[3:0]
This is table numbers which will be used by the Huffman decoder, if the selected instructions are the State machine instructions. However, if a ZRL comparator suits, the zrltab [3:0] field will be used, giving priority.
[1293]Probably, smtab [3:0] and zrltab [3:0] have the same value, when another table numbers do not need to be used according to whether ZRL conformity occurs. However, it notes that this may draw a strange simulation problem in Lsim. In the case of MPEG, there is no clear necessity of loading the register which directs the Huffman index number for ZRL (composition of only JPEG). However, these are still chosen, and when ZRL comparators are all that may be "strangeness" (therefore, it is not serious if which chosen), In spite of the fact that smtab [3:0] and zrltab [3:0] have the same value, a ZRL comparator will serve as "strangeness" and the following State will also become "strangeness."
zrltab[3:0]
It is table numbers for which they will be used by the Huffman decoder if the instructions as which this was chosen are the State machine instructions, and a ZRL comparator suits.
smvlc -- this is a VLC bit used by the Huffman decoder, if the selected instructions are the State machine instructions.
aluzrl[1:0]
This field controls the selection of the instructions sent to ALU. It may be a command from a parser (it memorized at the start of instruction sequence) State machine, or may be a command from the State machine. : [1294]
[Table 183]
alueob -- this wiring controls correction of the instructions sent to ALU based on an EOB comparator. This only forces zinput into ALU output mode. If this is output mode which are arbitrary choices and is separated from a certain;none, it is [anything] good. As for this, block - and a command word are sent to a token formatting part, and it controls suitable formatting of a data token there. : [1295]
[Table 184]
The remaining fields are the ALU instruction fields. These are appropriately documented in explanation of ALU.
[1296]B.2.2.5.4 "Correction of the Huffman State machine"
There is the necessity a data unit index part "knows" when the RUN portion of escape-code-ized Tcoefficient will be sent to a data unit index part in one mode of the State machine. Another approach has been used in order that it may avoid change of ROM, while this can be attained using the suitable bit in control ROM. The address included in ROM is supervised about this point, and the address value 5 is detected. This is a suitable place nominated in ROM which processes the RUN field. Of course, it will be obvious that it was able to do, if you think that he will program ROM so that

other selected address value may be used. Probably, the above-mentioned approach which uses the bit of control ROM has also been used.
[1297]B.2.2.6 "Outline guide tour"
In this invention, the Huffman decoder is called hd and hd actually contains a data unit index part (this is needed by restriction of the edited code generation). Therefore, hd includes the following main blocks.;
[1298]
[Table 185]
If the following explanation of the Huffman module is persons skilled in the art, it will be performed by overall explanation of various subsystem area which can be understood easily and which was shown in the detailed portion of a drawing.
[1299]B.2.2.6.1 "Explanation of "hd""
Three ports by which the logic for 2-wire-system interface control is usually controlled by a 2-wire-system interface: Include data input, data output, and a command. Besides, in addition, they are token valid which directs that :token with two "effective" lines from an input shifter is displayed on in data [7:0], and serial valid which directs that data is displayed on serial.
[1300]The most important signal generated is enabling which goes to latch. The most important thing is e1 which is enabling to ph1 latch. Although many of ph0 latches are not made possible, for these reasons, two enabling is provided and they are e0t relevant to e0 and token data relevant to serial data.
[1301]In this invention, a done signal (done0 and notdone0 of done, notdone(s), and those ph0 variants) directs the time of a primitive Huffman command being completed. done will be authorized at the time of completion of each primitive command including all the State machine commands, when the Huffman State machine command is executed. The signal notnew prevents the acceptance of the new command from a parser State machine until all the Huffman State machine commands are completed.
[1302]The control logic for the size fields is returned by the Huffman decoder during JPEG coefficient decoding about control of the information received from a data unit index part. This happens by two methods actually. If size is 1 correctly, this will be returned to exclusive signal notfbone0. Otherwise, it is returned from the output of a data unit index part. (It is directed that this generates out data [3:0] and signal fbvalid1.) The signal muxsize is produced in order to control multiplexing to the command register (sheet 10) of fed back data.
Besides, in addition, there is feedback by which the coefficient of 64 was decoded correctly. Since EOB is not coded in this case in JPEG, the signal forceeob is made. Similar, more, as mentioned above, two methods actually have this in carrying out with the signal for feeding backsize. Only the latch i-971 is loaded, and it is not cleared until a new parser State machine command is accepted, as data is fed back, when the usual feedback is performed (jpegeob). The signal forceeob is not actually made until a

Huffman code is decoded. Thus, although a fixed length code (getting it blocked size bit) is not influenced, the following Huffman code-ized information is replaced by the forced block end. Size is 1, and since only 1 bit is read when jpegeob0 is used, i-1255 and i-1256 delay a signal till exact time. Since the only signs in which what should be observed has zero size are EOB and ZRL, I hear that they cannot be occurred by zero size in this situation, and there are.
[1303]Decoding tcoeff tab0 (Huffman decoding using a Tcoeff table), It is quite random decoding of the command for making mba tab0 (Huffman decoding using an MBA table), and nop (no operation). There is a reason of some [generate / nop]. The fixed length code of size zero is one of them, a forceeob signal is also one of them (since any data should not be read from an input shifter even if an output is made, in order to sign EOB), and, finally table download nomination is the third reason.
[1304](It is made by FLC of size zero, and NOP) notfrczero guarantees that the result is zero, when NOP instructions are used. invert directs the time when the serial bit should be reversed before Huffman decoding (see section B.2.2.1.1). ring directs the time when the conversion factor ring should be applied (see the section B.2.2.1.2).
[1305]Decoding is further attained about the address of codes-per-bit ROM. These are built from small data path ROM. A signal is made into a duplex (for example, csha and csla). An address can be taken according to UPI access to the block chosen from a bit counter (bit [3:0]) or a microprocessor interface address (key addr [3:0]).
[1306]additional decoding -- JPEG tables (EOB, ZRL, etc.) -- although the Huffman index value of business is held, it is related with UPI read-out of a register [like]. The tri-state driver control these registers and for UPI read-out of codes-per-bit RAM is contained.
[1307]Operation data path decoding is also provided for the important specific number of bits. first bit is used in relation to the 1st coefficient trick of Tcoeff, and it is related to bit five applying a ring into a Tcoeff table. It takes notice of use of forceeob for simulating the operation which suits the index value by which the EOB comparator was decoded.
[1308]When a token is read from an input shifter about an extn bit, a related extn bit is read with it. Otherwise, the value of the last of extn is saved. This enables the test of the extn bit by a microcode program always, after a token is read.
[1309]When zerodat is authorized, it is forced so that top 4 bits of the Huffman output data may become zero. Since these only have effective value when carrying out decoding of the fixed length code, they are made into zero, when carrying out decoding of VLC and the token, or when NOP instructions are carried out for a certain reason.
[1310]A circuit detects the time of each command being completed, and emits a done signal. They are the; usual reason two groups are one of the reasons which are done intrinsically, and an exceptional reason. These are respectively processed by one of

the two 3 way multiplexers.
[1311]A low rank multiplexer (i-1275) processes the usual reason. In the case of FLC, the signal ndnflc is used. This is an output of a comparator [table numbers / bit counter]. In VLC, the signal ndnvlc is used. This is an output from an operation data path, and reflects the formula 9 directly. In the case of NOP instructions or a token, only one cycle is required, therefore a system is done unconditionally.
[1312]In this invention, the case where a higher rank multiplexer (i-1274) is exceptional is processed. When the decoder expects feedback of size in JPEG decoding (fbexpctd0), since only one bit is required, a decoder is done. It is done, when the decoder is processing the 1st bit of the first coefficient using a Tcoeff table and the bit zero of the present index are zero (see the section B.2.2.1.2). When neither of these conditions is filled, there is no exceptional reason for being done.
[1313]A NOR gate (i-1293) is made to transform a done state eventually. The state of producing by i-570 (it is got blocked and data is not effective) forces it done. This may be considered to be somewhat amusing. It is used after resetting fundamentally in order to make a machine be in a done state for preparation for the 1st command (done resets all the counters, registers, etc.).
[1314]When the signal notdonex detects an error, it is required. Since it will be anyhow forced done if an error is detected, the usual done signal cannot be used. Use of done will give the joint feedback loop.
[1315]Error detection and processing are performed by the circuit which detects all the possible error states. These [both] are 0Red in i-1190. In this case, i-1193, i-585, and i-584 constitute the Huffman error register of a triplet. When there is no "actual" error, i-1253 which makes an error impossible, and i-1254 are observed (see section B.2.2.3).
[1316]Besides, in addition, i-580 and i-579 provide the simple State machine which controls the acceptance of the first command after error detection with a related circuit.
[1317]As mentioned above, a control signal is delayed in order to double it with pipeline delay in a data unit index part and ALU.
[1318]Iotd bypass is a actual bypass signal sent to a data unit index part. It is corrected, when being controlled so that the Huffman State machine must bypass always, when a fixed length code is decoded.
[1319]Aluinstr [32] is a bit to which ALU makes a parser State machine be made to feed back (condition code). When the Huffman State machine is controlled, it is important for a signal to be recognized only once (whenever one of the primitive commands is completed, rather than a sake is carried out rather).
[1320]Aluinstr [36] is a bit at which ALU makes the step of a block counter value (when other ALU instruction bits specify increment). This must also be authorized only once.

[1321]Besides, in addition, these bits must be authorized only for the ALU instructions which output data to token conversion. Otherwise, the increment of the counter may be carried out before the output of the beginning to token conversion, and it may produce inaccurate value called cc in a data token.
[1322]In the mode which this invention illustrated, either alunode [1] or alunode [0] will become low, when ALU is outputted to a token formatting part.
[1323]Drawing 127 shows the Huffman State machine data path called hdstdp. There is UPI decoding for reading the output of Huffman State machine ROM.
[1324]Multiplexing is provided in order to process in the case of a location (see the section B2.2.4.6).
[1325]Correction of aluinstr [3:2] carries out processing which forces non-none the ALUoutsrc instruction field (see the explanation of the section B.2.2.5.3 and alueob).
[1326]Each bit of a command has a related multiplexer which makes selection between the possible sauce of a command about the command register for Huffman decoder block (x). : by which four control signals control this selection -- Selhold makes the present State hold to a register
[1327]Selnew makes a new command load from a parser State machine. This also makes possible further loading of the register which holds the command of the original parser State machine for next use.
[1328]Selold makes loading of a command start from the register holding the command of the original parser State machine.
[1329]/selsm makes loading of the command from Huffman State machine ROM cause.
[1330]Since table numbers are loaded also from the output data of a data unit index part in the case of table numbers, the situation is somewhat more complicated than it (selholdt and muxsize).
[1331]A latch holds the current address in Huffman State machine ROM. Logic detects which [of four possible commands] is performed. These signals are together put in order to form 2 bits of low ranks of a start address in the case of a new command.
[1332]Logic detects the time (since a command is usually a "simple" command) when the output of State machine ROM is still more meaningless. The signal notignorerom makes operation of the State machine impossible effectively, and makes impossible correction of the instructions sent to especially ALU.
[1333]The circuit which produces fixstate0 controls the jumping capability for this State machine to have been restricted.
[1334]Decoding is provided in order to drive away a signal to Huffman State machine ROM further. This is joint ROM of a data path style.
[1335]Generating of escape run is explained in the section B2.2.5.4.
[1336]Decoding prepares the register which holds the Huffman index number for signs,

such as ZRL and EOB. These registers can be loaded from UPI or a data path. center (decoding in es [4:0] and zs [3:0] is generating the select signal for the multiplexer which chooses which register or fixed value it is, in order to compare with a decoding Huffman index.)
[1337]Be related with the control logic for the Huffman State machines. Here, the "instruction" bit from Huffman State machine ROM is combined with various conditions in order to determine what you should next do and how the instruction word for ALU is corrected.
[1338]In this invention, the signals notnew and notold are used on the sheet 10 in order to control the operation of the Huffman decoder command register. They are made with the Huffman index comparator (neobmatch and nzrlmatch) here by a method clear from the control bit in State machine ROM (explained in section B.2.2.5.3).
[1339]The selection of the sauce for instructions sent to ALU is performed. Actual multiplexing is performed in the Huffman State machine data path hfstdp. Four control signals are generated.
[1340]Any one of aluseldmx (parser State machine instructions are chosen) or the aluselsm(s) (the Huffman State machine instructions are chosen) will be made when a block end is not encountered. Besides, in addition, it is corrected in order that the outsrc field of ALU instructions may push it aside to zinput.
[1341]A register holds the table numbers nominated during table download. Decoding is provided for codes-per-bit RAM. Additional decoding recognizes the time of signs, such as EOM and ZRL, downloading so that the Huffman index number register may be loaded automatically.
[1342]When a comparator reads FLC about a bit counter, it is detected when the bit of the number of the rights is read.
B.2.2.6.2 "Explanation of "hddp""
A comparator detects the special value of the Huffman index. A register holds the downloadable value for tables. A multiplexer (meob [7:0] and mzr [7:0]) chooses the value which should be used, and an exclusive-or gate and gating constitute a comparator.
[1343]An adder and a register are sections. The direct valuation of the formula explained in B.2.2.1 is carried out. It is thought that the further explanation is unnecessary here. Exclusive OR is used in order to reverse the data (i-807) explained in the section B.2.2.1.1.
[1344]A code register is 12 bit width. A multiplexing arrangement carries out ring substitution explained in the section B.2.2.1.2.
[1345]The Huffman index value is determined in ZRL and an EOB sign about the pipeline delay for data, and multiplexing between decoding-ized serial data (index [7:0]) and token data (ntoken0 [7:0]).

[1346]Codes-per-bit ROM and those multiplexing are used in order to determine the table which should be used. Since table selection information arrives behind time, this arrangement is used. The table of all after that is accessed and a right table is chosen.
[1347]The last multiplexing of codes-per-bitROM and the output of codes-per-bit RAM take place inside the block hdcpbram about Codes-per-bit RAM.
[1348]B.2.2.6.3 "Explanation of "hdstdp""
In this invention, Hdstdp comprises two modules. To a suitable pipeline stage, hdstdel is related to delaying a parser State machine control bit, for example until they are supplied to ALU and token conversion. It processes only the abbreviation half of the instruction word sent to ALU, and the remainder is processed by other modules hdstmod.
[1349]Hdstmod contains Huffman State machine ROM. Several bits of these instructions are used by the Huffman State machine control logic. The remaining bits are used in order to replace the portion which is not processed in hdstdel of an ALU instruction word (from a parser State machine).
[1350]Hdstmod is obvious and does not need explanation of what, either. There is only a pipeline delay register.
[1351]Hdstdel is also dramatically simple and it is processed by the multiplexer which corrects ROM and ALU instructions. The remainder of a circuit is related to UPI reading access to the half of the Huffman State machine ROM outputs. A buffer is also used for a control signal.
[1352]B.2.3 "Token FOMATTINGU"
Huffman decoder token FOMATTINGU by this invention is in the end part of the Huffman block. It is formatting the data from the Huffman decoder into a suitable token structure as the name shows the function. Multiplex transmission of the input data is carried out by the data in micro instruction word under control of a micro instruction word command field. The operation mode whose block is two; B.2.3.1 with data word and a data token "Micro instruction word"
[1353]
[Table 186]
B.2.3.2 "Operational mode"
[1354]
[Table 187]
B.2.3.2.1 "Data word"
In this mode, 8 bits of top of an input are sent to an output. 8 bits of partes basilaris ossis occipitalis are either 8 bits of partes basilaris ossis occipitalis of an input, the token field of micro instruction word or the both mixture according to a mask field. A mask is out data[16:8] =in data [16:8].
out data[7:0]=(Token[7:0]&(ff<<mask))in data[7:0]
The input bit number in a ** mix is expressed.

[1355]When a mask is set or more to 0x8, output data is equal to input data. This mode is used for the output word in a non-data token. If a mask is set as 0, out data [7:0] will serve as the token field of micro instruction word. Since the token header which does not contain any data is outputted, this mode is used. When a token header contains data, the number of data bits is given by the mask field.
[1356]When the exterior Extn (Ee) is set up, it becomes out extn=in extn, otherwise, is out extn=De.Bt, and Eb is don't care.
B.2.3.2.2 ″Data token″
It is used in order to format this mode with a data token, and there are two functions according to a signal and firstcoefficient. first coefficient is set up in reset. If the first data coefficient arrives with the micro instruction word which set cmd as one, out data [16:2] will be set as 0x1, and out data [1:0] will take the value of Bt field of micro instruction word. This is a header of a data token. If this word is accepted, the coefficient accompanied by a command will be loaded to the register RL, and first coefficient will take the value of Eb. If the following coefficient arrives, out data [16:0] will take the coefficient before RL memorized. When encountering a block end, Eb is set up, firstcoefficient is set up, and this is the following data token (first coefficient), i.e., If.

```
{out data[16:2] = 0x1 out data[1:0] = Bt[1:0]
RL[16:0] = in data[16:0]
}
else {out data[16:0] = RL[16:0]
RL[16:0] = in data[16:0]
}
```

It ensures that preparation of out extn = −Eb is completed.
[1357]B.2.3.3 ″Notes″
According to this invention, most instruction bits are supplied by the parser State machine by the usual method. However, the two fields are actually supplied by other circuits. Direct continuation of the above-mentioned Bt field is carried out to the output of an ALU block. These two bit fields give the present value of cc or color component. In this way, if a data token header is built, 2 bits of the lowest order will take out a color component from an ALU counter directly. When block − and the sign id are decoded, Eb bit is authorized [2nd] in the Huffman decoder always (or since the coefficient of the last within a block is coded, when it is JPEG 1 is assumed to be).
[1358]An in extn signal is pulled out in the Huffman decoder. About a token, it only has a meaning, when an extension bit is supplied with a token word by the usual method.
[1359]B.2.4 ″Parser State machine″
The parser State machine of this invention is an actually very simple circuit piece. Programming of microcode ROM is complicated and it is a section. It discusses in B.2.5.

[1360]Intrinsically, a machine comprises a register holding the current address. The lookup of this address is carried out in microcode ROM, and it makes a micro code word. The increment of the address is carried out in a simple incrementor, and this address by which increment was carried out is one of the two possible addresses used for the following State. The address of another side is the field in the microcode ROM itself. Thus, each instructions are jump instructions potentially and can be jumped for the location specified in a program. If a jump is not performed, control progresses to the next location in ROM.
[1361]A series of condition code bits of 16 are provided. One of the condition of such is chosen (field in microcode ROM), and, in addition to it, it may be reversed (too bit in microcode ROM). The signal produced as a result chooses either the address by which increment was carried out, or the jump address in microcode ROM. Since one of the condition evaluates as False, it is wired. A jump will not be generated if this condition is chosen. Or instead, this condition is chosen and reversed and a jump, i.e., an unconditional jump, always occurs.
[1362]
[Table 188]
B.2.4.1 ″2-wire system interface control″
The 2-wire system interface control by this invention is an exception for a while in this block. A 2-wire system interface is between a parser State machine and the Huffman decoder. This is used in order to control advance of a command. It will wait for the parser State machine until a predetermined command is accepted, before moving forward in order to read the following command from ROM. Besides, in addition, a condition code is returned through the wire from ALU.
[1363]Each command has a bit in microcode ROM, and it enables it to specify what should be waited for feedback. When this occurs, a new command is not displayed after the instructions were accepted by the Huffman decoder until the feedback wire from ALU is authorized. This wire and fb valid direct that the condition code supplied by ALU now is effective in the meaning of reflecting the data relevant to the command which they urge to wait for feedback.
[1364]According to this invention, the feature is used when building the condition jump command which determines the next State for jumping to special data pieces as a result of decoding (or processing). Condition depending on the data in a pipeline could not be tested without these facilities, since it means that the time of a two wire control of a specific command reaching a predetermined processing block (getting it blocked in this case ALU) is uncertain.
[1365]All instructions are not sent to the Huffman decoder. A certain instructions of a data pipeline are unnecessary, and can be performed. These tend to be jump instructions. The bit in microcode ROM chooses whether instructions are displayed on the Huffman decoder. Otherwise, even if the Huffman decoder did not need to

accept instructions, therefore the pipeline has damped, execution can be continued in such a situation.
[1366]B.2.4.2 "Event processing"
Two event bits are placed into a parser State machine. One is called the Huffman event and other one is called a parser event.
[1367]The parser event is the simplest. The "condition" supervised by this event is only a bit in microcode ROM. Thus, instructions can produce a parser event by setting up this bit. The instructions which perform this write a suitable constant in a rom control register, and an interrupt service routine enables it to judge the cause of interruption typically.
[1368]After serving a parser event (promptly [when / or / mask out of the event is carried out]), control is recovered on the stopped point. When the instructions which made the event start have jump instructions (it is estimated that the state is a genuine article), a jump is performed by the usual method. Therefore, it can jump on an error hair drier after service by coding a jump.
[1369]The Huffman event differs from it. The state where it is supervised is OR of three Huffman error bits. Actually, this state is processed to a parser event by a very simple method. However, the additional wire from the Huffman decoder and huffintrpt are authorized always, when an error occurs. This makes an error hair drier jump control in a microcode program.
[1370]Therefore, when the Huffman error occurs, a sequence interrupts and includes generating and a block stop. Control is transmitted to an error hair drier after Sir BISHINGU. It is impossible for there to be no call mechanism and to return at the point in a microcode before the error generated following error processing unlike the usual interruption.
[1371]huffintrpt can be authorized without the Huffman error occurring. This is a section. As it discussed in B.2.2.3, it generates, when a no-error error is special. In this case, any interruption is not generated (to microprocessor interface), but control is sent to an error hair drier (inside of a microcode). Since the Huffman error register is clear in this case, a microcode error hair drier is in such a situation.
It can determine to answer according to it.

[1372]B.2.4.3 "Special location"
There are some special locations in microcode ROM. Four locations of the beginning in ROM are the entry points to a main program. Control progresses to one of the four locations of these at the time of reset. The location which should be jumped is determined by an ALU register and the coding standard chosen in coding std. Since this location is reset by zero in itself by true reset, control progresses to location zero. However, only a parser State machine is resettable using the UPI register bit CED H TRACE RST in CED H TRACE. In this case, a coding std register is not reset but

control progresses to one with the first four appropriate locations.
[1373]The 2nd four location (0x004 to 0x007) is used when Huffman interruption takes place. Typically, a actual error hair drier is put on each of these locations. Here, it succeeds in selection of a location as a result of a coding standard.
[1374]B.2.4.4 ″Tracing″
A trace mechanism is mounted as help on diagnosis. A microcodeis [single-step-] made toize this. The bits CED H TRACE EVENT and CED H TRACE MASK in the register CED H TRACE control this. As the name shows, they act to the usual event bit by the method alike very well. However, the group division of them is not carried out together with other event bits by differences among some (UPI interruption should not arise in particular).
[1375]A tracing mechanism is made one when CED H TRACE MASK is set as one. After each microcode instructions are read from ROM, however before it is displayed on the Huffman decoder, a trace event occurs. In this case, CED H TRACE EVENT is set to one. Since any interruption is not produced, it must be registered. All the micro code words can use in register CED H KEY DMX WORD 0 - CED H KEY DMXWORD 9. If those instructions are required, they are correctable at this time. If one is written in CED H TRACE EVENT, the instructions will be performed and CED H TRACE EVENT will be cleared. The following micro code word which should be carried out is immediately read from ROM after this, and a new trace event occurs.
[1376]B.2.5 ″Microcode″
Although a microcode is programmed using the assembler hpp, it is a very simple tool and the great portion of sampling is performed using a macro preprocessor. The standard C preprocessor cpp is used for this purpose.
[1377]A code is directed as follows.
[1378]Ucode.u is the main file. First, this contains tokens.h which defines a token. Next, regfile.h defines the ALU register map. fields.u provides the symbol list which defined various fields in a micro code word, and was defined for each possible bit pattern in the field. Next, the label used in a code is defined. After this step, in order to define cppmacros of a large number which clarify fundamental instructions, instr.u is contained. Next, errors.h defines the number which defines a parser event. Next, unword.u defines the order on which the field is put, in order to build a micro code word.
[1379]The remaining ucode.u is the microcode program itself.
[1380]B.2.5.1 ″Instructions″
This section explains various instructions defined in ucode.u. Many cases concerning one theme (especially ALU instructions), since there is little modification, explanation about all the instructions is not given here.
[1381]B.2.5.1.1 ″Huffman and data index instructions″
In this invention, the Huffman decoder uses H NOP instructions. It is no operation

instructions. In the meaning that any data is not decoded, Huffman does nothing. The data made by these instructions is always zero. Therefore, related instructions are sent to ALU.
[1382]the next instructions -- token group; -- they are H TOKSRCH, H TOKSKIP PAD, H TOKSKIP JPAD, H TOKPASS, and H TOKREAD. These all read the token from an input shifter, and send them to the remaining machines. H TOKREAD reads one token word. H TOKPASS can be used in order to read all the tokens to a zero extn bit. A related command is repeated for each word of a token. H TOKSRCH throws away all the serial data in front of a token, and reads one token word. H TOKSKIP PAD flies padding bits (H.261 and MPEG), and reads one token word. H TOKSKIP JPAD carries out that it is the same for JPEG padding.
[1383]H FLC (NB) reads the fixed length code of NB bit.
[1384]H VLC (TBL) reads vic using the table directed (sent as mnemonic (tcoeff), for example, H VLC).
[1385]H Although FLC IE (NB) is the same as that of HFLC, an ignore errors bit is set up.
[1386]H Although TEST VLC (TBL) is the same as that of H FLC, a bypass bit is set up so that the Huffman index may be sent through an uncorrected data unit index part.
[1387]H FWD R and H BWD R read FLC of the size respectively directed with the ALU registers r fwd r size and rbwd r size.
[1388]H DCJ reads the DC coefficient of a JPEG style, and the table numbers from ALU.
[1389]H TC/OE/FF and H DCTC/OE/FF read a conversion factor. In HDCTC/OE/FF, the 1st coeff bit is set up, and while it is a thing for a non-intra block, it is because H TC/OE/FF is the intra block after DC term was read.
[1390]H NOMINATE (TBL) nominates the table for the next download.
[1391]H DNL (NB) reads NB bit and downloads them on the nominated table.
[1392]B.2.5.1.2 "ALU instructions"
Explaining in detail actually has too much many ALU instructions too much. The fundamental method by which Mnemonics construction is carried out is discussed, and, thereby, instructions can be read now. If there are these by one of persons skilled in the art, they must be what can be understood easily.
[1393]General "load" instructions are used with regards to many ALU instructions moving data to a place from a place therefore. In Mnemonic, the contents of y are loaded to x, i.e., a destination is recorded previously and, as for it, A LDxy turns out that sauce comes to the next.
[1394]
[Table 189]
As an example, LDAI loads the data from the data input port of ALU to A register. If an ALU register file is specified, Nima Nick will take the address so that LDAF (RA) may

load the contents of location RA in a register file to A.
[1395]ALU has the capability to correct data as it is moved from sauce to a destination. In this case, arithmetic is directed as some source data. Therefore, Mnemonic LDA AADDF (RA) loads contents plus of the present of A register, and the contents of the directed location in a register file to A. Another example is LDA ISGXR, and it takes input data, and a sign is extended from the bit directed in a RUN register, and memorizes the result to A register.
[1396]In many cases, one or more destinations are specified for the same result. Here, by LDF LDA ASUBC (RA), the result of A minus constant is loaded to both A register and a register file as an example.
[1397]Other Nima Knicks exists for special operation. For example, it is used in order that CLRA may clear A register, and RMBC is used in order to reset a macro block counter. These are quite obvious and are explained in the comment in instr.u.
[1398]One exception is using the suffix O, in order to direct that the result of operation is outputted to token FOMATTINGU in addition to the usual operation. Thus, LDFI O (RA) memorizes input data, and also sends it to token FOMATTINGU. Or instead, if it is a request, this may be LDF LDO I (RA).
B.2.5.1.3 ″Token FOMATTINGU instructions″
This is T NOP ″no operation″ instructions. Since it is impossible actually to build no operation instructions, this is incorrect **. However, since ALU is not outputted to token FOMATTINGU, this is used when the instructions are not important.
[1399]T TOK outputs a token word.
[1400]T DAT outputs a data token (accepted and used with the Huffman State machine instructions) word.
[1401]T GENT8 produces a token word based on 8 bits of the fixed field.
[1402]T An extension bit is 1 although GENT8E is the same as that of T GENT8. T In the remainder of the bit which comes from the fixed field, OPD (NB) is NB bit of the data from the pars-basilaris-ossis-occipitalis NB bit of an output.
[1403]T An extension bit is high although OPDE (NB) is the same as that of T OPD.
[1404]T OPD8 is a short hand for T OPD (8).
[1405]T OPD8E is a short hand for T OPDE (8).
[1406]B.2.5.1.4 ″Parser State machine instructions″
These instructions and D NOP increase no operation, i.e., an address, ordinarily, and a parser State machine does no special things. The remainder of instructions is sent to a data pipeline. Standby is not generated.
[1407]D Although WAIT is the same as that of D NOP, wait for feedback to occur.
[1408]A simple jump group. D Nima Knicks like JMP (ADDR) and D JNX (ADDR) will jump, if conditions are fulfilled. Instructions are not outputted to the Huffman decoder.
[1409]External jump group. D Nima Knicks like XJMP (ADDR) and D XJNX (ADDR). Although these are the same as that of the above-mentioned simple counterpart, the

instructions are outputted to the Huffman decoder.
[1410]A jump and a standby group. D Nima Knicks, such as WJNZ (ADDR). These instructions are outputted to the Huffman decoder, and a parser waits for the feedback from ALU, before evaluating a state.
[1411]The following Nima Knicks is used for the state itself.
[1412]
[Table 190]
D EVENT causes generating of an event.
[1413]D DELT is for construction of default instructions. This causes an event and jumps it for a location with the label dflt. Since these instructions are used in order to fill ROM, and the trap of the intact location is carried out, don't perform.
[1414]D ERROR causes an event and jumps it on the label srch dispatch assumed to plan recovery from an error.
Section B.3 "Huffman decoder ALU"
B.3.1 "Preface"
According to this invention, a Huffman decoder ALU subblock provides the arithmetic and logical functionality [being general] for the Huffman decoder block. It has the capability to perform addition and subtraction operation, various sign extended operation of a type, and formatting to run-sign-level TORIPURUZU of input data. It has a flexible structure specified under 2-wire system interface control by the micro instruction word which reaches ALU as synchronous [the exact operation and composition] as input data that is,.
[1415]In addition to 36-bit instructions and 12 bit-data input port, ALU has a 6-bit run port, and (it is on a token bus actually) an 8-bit fixed port. These all drive the bus of each width through an ALU data path except for micro instruction word. An extension bit is expressed and there is 1 bit in the micro instruction word outputted with a 17-bit run sign level (out data). There are a series of condition codes outputted to each end of an ALU data path with a 2-wire system interface, the effective signal of these selves, and cc valid. Other Huffman decoder subblocks and microprocessor interfaces have an accessible register file via ALU.
[1416]B.3.2.2 "Basic structure"
The basic structure of Huffman ALU is as being shown in Drawing 135. : in which it includes the following components -- the Run register (6 bits) provided with A register sauce multiplexing provided with input block output block condition code BUROKKUSOSU multiplexing
the sign extension logic register file (except for an output block) provided with an adder / subtractor sauce multiplexing provided with sauce multiplexing -- each of a block of these the output on the bus which runs through a data path, [push aside and] These buses are used as an input to multiplexing for block sauce next. For example, an adder has a data path bus of itself which is one of the possible inputs to A

register. Similarly, A register has a bus of itself which forms one of the possible inputs to an adder. As it wrote clearly in the section 7 about micro instruction word at this point, only the subset of all the possibilities exists.
[1417]In a single cycle, it is possible to perform either instructions of the Addo base or sign extension base instructions. It restricts, when those operations are strictly parallel, and even if it performs those both in a single cycle, it does not interfere. If it puts in another way, it will add and sign-extend or sign extend, and the sequence of addition will not be allowed. The both are not made although it is written in whether a register file is read in a single cycle.
[1418]: in which output data has the three fields - run - 6-bit sign - 1 bit levels - When 10 bit data are straight sent through ALU, 11 bits with least importance of an input data register are latched to a sign and the level field.
[1419]The multicycle operation of restricted ALU is programmable. At this point, the required number of cycles is decided by the contents of the register file location, that address is specified in a microinstruction, and while repetitive counters decrease in number even to 1, repeat execution of the same operation is carried out. Typically, this equipment is used in order to shift the left, it adds A register to itself using an adder, and that result is returned to A register and it memorizes it.
[1420]B.3.3 "Adder / subtractor subblock"
This is an adder of 12 bit width and has the arbitrary reversal to the input2, and arbitrary setting of a carry-in bit. An output is the 12-bit sum total and carry-out is not used. : and ADD to which the seven modes are in operation: :input1 which adds the carry of a set to zero +input2 and ADC: :input1 which adds the carry of a set to 1 +input2+1 and SBC:input2 are reversed, It is the carry of a set to zero :input1 - input2 - 1 and SUB:input2 are reversed and it is the carry of a set to 1 :input1 If it is - input2 and TCI:input 2< 0, SUB will be used, otherwise, ADD will be used. This is used with input1 set as zero, in order to obtain magnitude value from a two's complement value.
[1421]DCD(DC difference): If it is input2<0, ADC will be performed, otherwise, ADD will be performed.
[1422]VRA(vector remainder Addo): If it is input1<0, ADC will be performed, otherwise, SBC will be performed.
[1423]B.3.4 "Sign extension subblock"
This is a 12-bit unit to which a sign extends the input data from a size input in various modes. Size is the 4-bit value to 0-11 (11 is related with the most important bit about the bit which is the least important for 0). An output is 12-bit correct data value, and is a "sign" bit.
[1424]In SGXMODE=NORMAL, all the bits more than a size-th bit take the value of a size-th bit. Not all the following change but remain as they are. A sign takes the value of a size-th bit. for example, : -- data = 1010 1010 1010 size = 2 output =0000 0000 In 0010 and sign=0 SGXMOD=INVERSE. While all the bits more than a size-th bit take

the reciprocal of a size-th bit, not all the following change but remain as they are. A sign takes the reciprocal of a size-th bit. for example, : -- data = 1010 10101010 size = 0 output = 1111 1111 In 1111 and sign=1 SGXMODE=DIFMAG. if a size-th bit is zero, while all the bits below a size-th bit will be reversed, no above-mentioned things change but come out as it is. If a size-th bit is 1, all the bits will remain without changing. In both cases, a sign takes the reciprocal of a size-th bit. This is used in order to obtain the magnitude of AC difference value. for example, : -- data = 0000 1010. 1010 size = 2 output=. 0000 1010 1101 and sign=1 data = 0000 1010 1010 size = 1 output = 0000 1010 In 1101 and sign=0 SGXMODE=DIFCOMP. the bit of all (it is -- although -- a size-th bit is not included) more than a size-th bit takes the reciprocal of a size-th bit, and, on the other hand, all the bits not more than it remain, without changing. A sign takes the reciprocal of a size-th bit. This is used in order to take two complement values for DC difference value. for example, : -- data = 1010 1010 1010 size= 0 output = 1111 1111 1110 and sign=1B.3.5 ″Condition code″
There are two bytes (16 bits) of the condition code which the Huffman block uses, among those a specific bit is made by ALU/register file. These are a sign condition code, a zero condition code, an extended condition code, and a change detection bit. Since two of the last of these codes are not used by the parser like other things, they are not a condition code actually.
[1425]it is updated, when issuing instructions so that a parser may update, and as for a sign, zero, and an extended condition code, a condition code valid signal is sent by a high pulse only once for instructions of these each.
[1426]A sign condition code is a sign extension sign output only latched, and on the other hand, a zero condition code is set as 1, when the input to A register is zero. An extended condition code is an input extension bit latched regardless of OUTSRC.
[1427]: and the result which may be used in order that a condition code may evaluate a specific condition type are equal to a constant. - subTORAKUTO is used, Zero condition and a result are equal to a register value. - subTORAKUTO is used, A zero condition register is equal to a constant. - subTORAKUTO is used, Zero condition register bits set - Sign extension is used, It is a bits set as a result of sign condition. - sign extension is used, It notes that it is only possible to evaluate one specific bit rather than evaluating a multibit like [in the case of the conventional logic AND], when using the combination of sign condition sign extension and a sign condition code.
[1428]In this invention, although a change detection bit is made using the logic same as a zero condition code, it does not have an effective related signal. The bit in a microinstruction differs from that in which the value written in the register file recently has already existed (two clock cycles are required and). When it means setting REG-MODE as READ first and then setting REGMODE as WRITE, it is directed that a change detection bit should be updated. Next, interruption of a microprocessor is started when the changed value is detected. Although a change

detection bit is reset by starting Change Detect by the usual method, REGMODE is set as READ.

[1429](Refer to the following which forms a part of register file) :Mb Start, Pattern Code, Restart, and Pic Start in which hard-wired macro block counter structure also makes the following condition codes.

[1430]B.3.6 "Register file"

The address map for register files is shown below. It uses an ALU data path and 7 bit-address space common to both UPI(s). Many locations cannot be accessed by ALU, but these are the counters in hard-wired macro block structure typically, and are the registers in ALU itself. Although the latter register has access for exclusive use, some address maps for UPI are formed. (Oversize is displayed by 0 in a table) The location of a certain multibyte has one ALU address and multiplex UPI address. Similarly, the register group, as for, an index is carried out by an ingredient count and CC (directed by I in a table) is processed by ALU as one location. This makes microprograming easy for [for initial setting and resetting] the operation of a block level.

[1431]except for an exclusive ALU register (UPI -- read-only), all the locations are read/write and all the counters are reset by the bit within an instruction word at zero. A pattern code register has the capability of a rightward shift, and the least important bit forms a Pattern Code condition bit. All the registers of hard-wired macro block structure are displayed in [M] front, and also notes called Cn in what is a counter (n-bit) are attached.

[1432]The portion of everything [location / specific in this invention] but the Huffman subsystem coding standard, It has contents by which hard-wired one is carried out to one location (1 bit word) for [each] two r-size locations, the ac huff table to the Huffman decoder, and a dc huff table.

[1433]The address of a bold letter shows that it is [location] accessible in both ALU and UPI, otherwise, they have only UPI access. Data token ****** which chooses the physical location in the group in whom the register group which is not directed by ALU through CC can have one ALU address specified in an instruction word, and whom CC should access. Although the first address should have been conventionally used for the ALU address, it may be a thing of which register in a group. Whichever an address this is, it is enough, but the address of the lower one is used among the addresses of a pair, and, also in the case of the location of the multibyte which should be accessed, it is actually applied. The locations 2E and 2F not only let a keyhole register pass, but take notice of [in / that is, / a top-level address map (displayed by T)] an accessible thing. These two locations are also reset by zero.

[1434]In order that a register file may improve an access speed, it is physically divided by four "banks", but this does not affect addressing by any methods. The main table shows the allocation for MPEG and provides respectively the variation for JPEG and

H.261 in two sections repeated.
[1435]
[Table 191]
[1436]
[Table 192]
[1437]
[Table 193]
[1438]
[Table 194]
[1439]
[Table 195]
[1440]
[Table 196]
[1441]
[Table 197]
[1442]
[Table 198]
[1443]
[Table 199]
[1444]
[Table 200]
B.3.7 ″Micro instruction word″
According to this invention, ALU micro instruction word is divided into many fields, and the each controls the aspect of affairs where the above-mentioned structures differ. The total number of bits used in an instruction word is 36, for (extension bit input, the minimum encoding which crosses plus 1) and the field is adopted, and the pliability of hardware constitutions at its maximum is maintained. An instruction word is divided so that it may explain in full detail below. What does not change the State of default field value, i.e., ALU, or a register file is shown by italic type.
[1445]
[Table 201]
[1446]
[Table 202]
[1447]
[Table 203]
[1448]
[Table 204]
[1449]
[Table 205]
Section B.4 ″Buffer manager″

B.4.1 ″Preface″
This writing explains the purpose of the buffer manager by this invention, operation, and mounting (bman).
[1450]B.4.2 ″View″
A buffer manager provides a DRAM interface with four addresses. These addresses are page addresses in DRAM. A DRAM interface maintains two FIFO, a coding data buffer, and a token data buffer in DRAM. Therefore, there are the read address and write address for each buffer for four addresses.
[1451]B.4.3 ″Interface″
A buffer manager is connected with a DRAM interface only to a microprocessor. It is necessary to use a microprocessor only in order to set up the ″initial-setting register″ shown in B.4.4. The interfaces provided with the DRAM interface are four bit addresses [18] controlled by a REQuest/ACKnowledge protocol for each address. (Since there is no buffer manager in a data path, the buffer manager lacks the 2-wire system interface.)
A buffer manager turns OFF a DRAM interface clock generation machine, and makes a DRAM interface scan chain one.
[1452]B.4.4 ″Address computation″
:-initial-setting register in which the read address and write address for each buffers are made from nine bit registers [18] (RW from a microprocessor)
- The base address, the LENGTHCB - maximum size (it can set to the page of a coding data buffer), and BASETB of a BASECB - coding data buffer - The base address LENGTHTB of a token data buffer. - The size (it can set to page) dynamic register of the maximum size (it can set to a page) and LIMIT - DRAM of a token data buffer (RO from a microprocessor)
- READCB. - The token data buffer read pointer and NUMBERTB about the coding data buffer write pointer and READTB - BASETB about the coding data buffer read pointer and NUMBERCB - READCB about BASECB. - For calculating the token data buffer write pointer address about READTB, it is :-readaddr=(BASE+READ) mod LIMITwriteaddr= (READ+NUMBER) (mod.). Since a LENGTH+BA SEmod LIMIT buffer may wrap the surroundings of DRAM, the term of mod LIMIT is used.
[1453]B.4.5 ″Explanation of a block″
A buffer manager is constituted from this invention by the module of three top levels connected in the ring in which SUNOPA supervises a DRAM interface joined part. Modules are bmprtize (priority) and bminstr (instructions), bmrecalc (re-calculation) is arranged in the ring of the order, and omsnoop (SUNOPA) is arranged on address output.
[1454]A module and Bmprtize process a REQ/ACK protocol and the FULL/EMPTY flag for buffers, and maintain the State of each address, i.e., isit avalid address?. From this information, bminstr is ordered which (in a certain case) address for it to

re-calculate. It operates the BUF CSR (status) microprocessor register in which a FULL/EMPTY flag is shown, The buf access microprocessor register which controls microprocessor right access to a buffer manager register is operated.
[1455]If they order calculating an address by bmprtize, a module and Bminstr issue six instructions (one per two cycle), and in order to calculate an address, they will control bmrecalc.
[1456]A module and Bmrecalc re-calculate an address under instructions of bminstr. It moves instructions for every two cycle, and contains simple ALU which can do all the initial-setting registers, dynamic registers and addition, subtraction, and a modulus. It will calculate an address, if Sbmprtize of the FULL/EMPTY State which it detects is told and it is completed.
[1457]B.4.6 "Block mounting"
B.4.6.1 "Bmprtize"
In reset, a buf access microprocessor register is set as 1, and enables setting out of an initial-setting register. While buf access has returned one, any address computation is not started. It is because those calculations are meaningless without an effective initial-setting register.
[1458]buf access carries out full authorization once -- having (zero are written in it) -- since the purpose is to validate all the four addresses, mbprtize is enacted so that all (re-calculating them) the address may be validated. In this stage, a buffer manager is "a starting rise" (it is got blocked and no addresses are calculated yet), and any requests are not authorized in this way. Once all the addresses become effective, start-up will be completed and all the requests will be authorized. this point in time to an address -- invalid -- becoming (since it is used once and recognized) -- it will be re-calculated.
[1459]It is not required to attach a priority between addresses. While a buffer manager can re-calculate an address every 12 cycles, it is because the DRAM interface can use an address every 17 cycles in the highest case. Therefore, only one address becomes invalid at once after start-up. Therefore, bmprtize will re-calculate the invalid address which is not calculated now.
[1460]In an invention, since start-up is reinputted whenever buf access is authorized, any addresses are not supplied to a DRAM interface during micro-processor access.
[1461]B.4.6.2 "Bminstr"
A module and Bminstr contain MOD12 cycle counter (the number of cycles which it takes in order to make an address). It notes that an odd number cycle ends instructions to even numbered cycles starting instructions. For :read address decoded as follows with whether it is a lead or light calculation by top 3 bits of instructions for bmrecalc : [1462]
[Table 206]
For a write address : [1463]

[Table 207]
Notes: The result of the last operation is always held in an accumulator.
[1464]When there is no address which should be re-calculated, a cycle counter is raced by zero and the instructions written in neither of a register in this way are produced. This does not have any influences.
[1465]B.4.6.3 "Bmrecalc"
A module and Bmrecalc carry out one operation every two clock cycles. It is latched in the instructions from bminstr (and the buffer and io type) on an even number counter cycle (start alu cyc), and latches the operation result on an odd number counter cycle (end alu cyc). In addition to the register which instructions specify, an operation result is always memorized in an Accum register. About whether use of the address calculated now has passed uses whether bmrecalc makes a buffer full and empty on end alu cyc, It notifies to bmprtize about when an address and full/empty are calculated well (load addr).
[1466]Full/empty is calculated using the sign binary digit of an operation result.
[1467]Although modulus operation is not a true modulus, A mod B is carried out as follows. : (A>B?(A-B):A)
However, in the following case, it is only wrong, and this is A> (2B-1).
This never happens.
[1468]B.4.6.4 "Bmsnoop"
A module and Bmsnoop comprise four 18-bit super SUNOPA which supervises the address supplied to a DRAM interface. SUNOPA must be super in order to enable the test of external DRAM on chip (it is got blocked and must be able to access by clock running). These SUNOPA must act on a REQ/ACK system, therefore it differs from other things on a device.
[1469]REQ/ACK is used on this interface as what opposes a 2-wire system protocol. It is because what information is returned for to a sender (that is, a receipt is told) is indispensable and acceptance does not carry out it. Therefore, this supervises a FIFO pointer strictly.
[1470]B.4.7 "Register"
In order to obtain the right access of the microprocessor to an initial register, one should be written in bufaccess, and access is obtained when buf access reads one over. On the contrary, in order to abandon the right access of a microprocessor, zero should be written in buf access. Access is obtained when bufaccess reads zero over. It notes that buf access is reset by one.
[1471]Although the dynamic register and initial-setting register of this invention can be read at any time, right access must be obtained in order to guarantee that the dynamic register is not changing the microprocessor.
[1472]It has intention of an initial-setting register being written in only once. A buffer is made made a mistake in and operated by rewriting them. However, making the

length of an on-the-fly buffer increase, and making a buffer manager use new length for timely is planned. In order that the value in an initial-setting register may look at whether it can sense that a buffer does not overlap for example, any checks are not performed until now. A user is to blame for this.
[1473]
[Table 208]
D points to a register bit among front, and any register bits do not show x.
[1474]
[Table 209]
[1475]
[Table 210]
[1476]
[Table 211]
B.4.8 ″Collation″
The check placed small FIFO's, after a straw man interfaced [DRAM], and in Lsim, it was carried out in C-code as a part of top-level chip simulation.
[1477]B.4.9 ″Test″
through SUNOPA in bmsnoop, and a dynamic register (B. it was shown in 4.4), the test cover ridge to bman comes out, and there is, and it is carried out using the scan chain which is a part of DRAM interface scan chain.
Section B.5 ″Reverse modeler″
B.5.1 ″Preface″
This writing explains the reverse modeler (imodel) by this invention and the purpose of token FOMATTINGU (hsppk), operation, and execution.
[1478]Notes: Although hsppk is a hierarchical portion of the Huffman decoder, it is a part of reverse modeler functionally. Therefore, it is better to discuss with this section.
[1479]B.5.2 ″View″
Token buffers are between imodel and hsppk and can include all a lot of data in off-chip DRAM. In order to ensure efficient use of this memory, data must be a 16-bit format. Token FOMATTINGU ″packs″ the data from the Huffman decoder to this format for token buffers. Therefore, a reverse modeler ″takes out″ (unpacking) out data from a token-buffers format.
[1480]However, the main functions of a reverse modeler are zero data's running from a run / ″level″ code, and extending to a level. it -- in addition, it guarantees that a reverse modeler provides the ″gate″ for stopping the stream which a data token has a coefficient of at least 64, and is not filling start-up criteria.
[1481]B.5.3 ″Interface″
B.5.3.1 ″Hsppk″
In this invention, Hsppk has token buffers as the Huffman decoder and an output as an input. Both interfaces are 2-wire system type things, an input is a 17-bit token port,

and an output is a 16-bit "data packed" plus FLUSH signal. Besides, in addition, since the clock of the Hsppk is carried out from the Huffman clock generation machine, it is connected to the Huffman scan chain.
B.5.3.2 "Imodel"
Imodel has token-buffers start-up output gate logic (bsogl) as an input, and has an inverse quantization device as an output. The input from token buffers is a 16-bit "data packed" plus block end signal, and the input from bsogl is one wirestream enable. An output is an 11-bit token port. All the interfaces are controlled by a 2-wire system protocol. Imodel has the clock generation machine and scan chain of itself.
[1482]In the output, both blocks have micro-processor access only to SUNOPA.
[1483]B.5.4 "Explanation of a block"
B.5.4.1 "Hsppk"
Hsppk receives 17 bit data from Huffman, and outputs 16 bit data to token buffers. The split of whether this omits input data to 12 bit word first is carried out, and it is attained by next packing these words to a 16-bit format.
[1484]B.5.4.1.1 "Splitting"
Hsppk receives 17 bit data from reverse Huffman. This data is formatted into 17 bits using the following formats.
[1485]F specifies a format and, as for extension bit;R, length bit (sign magnitude showed) or non-data token bit;x of run bit;L is [;E] don't care.
[1486]
A FLLLLLLLLLLLFormat 0ELLLLLLLLLLLFormat 0aFRRRRRR00000Format 1 usual token only occupies 12 bits of partes basilaris ossis occipitalis, : which takes the following gestalten -- ExxxxxxLLLLLLLLLLL -- however, A data token in each word with the following gestalten a run and a level. : which it has -- ERRRRRRLLLLLLLLLLL -- :ERRRRRRLLLLLLLLLLL->FRRRRRR00000 Format 1 from which this is divided into the following format, and ELLLLLLLLLLLFormat 0a -- or, If a run is a zero format, 0 is used, an extension bit is lost in the :E000000LLLLLLLLLLL->FLLLLLLLLLLL Format0 format 0, and it turns out that it is assumed that it is 1. Therefore, it cannot be used when extension is zero. In this case, the format 1 can be used unconditionally.
[1487]B.5.4.1.2 "Packing"
All the data word is 12 bit width after splitting. All the four 12 bit word "is packed" to three 16 bit word. : [1488]
[Table 212]
B.5.4.1.3 "Flashing of a buffer"
The DRAM interface of this invention collects a block and the 16-bit "finishing [a pack]" words of 32, and writes them in a buffer. This means that data can be stuck into a DRAM interface in the end of a stream, when the block is not completed selectively. Therefore, a flashing mechanism is required. Therefore, .Hsppk sends a

signal to a DRAM interface so that the present partial completion block may be written in unconditionally.
[1489]B.5.4.2.1 "Imup (unpacker)"
: in which Imup achieves three functions -- four -- the data from the 16-bit format is unpacked to 12 bit word.
[1490]
[Table 213]
5) Maintain the right data between flashing of token buffers.
[1491]When a DRAM interface carries out a flash plate by writing in the present partial completion block unconditionally, the data which is not helpful remains in a block. Imup must delete the data which is not helpful to the end of a block, i.e., all the data from a flash plate token.
[1492]6) Keep back data until start-up criteria are filled. The data output from a block is conditional [are accepted from "effective" (stream enable) buffer start-up for each different stream]. Therefore, 12 bit data are outputted to hsppk.
B.5.4.2.2 "Imex" (EXpander)
In this invention, Imex extends all the run length codes to the run of zero, and the level following it.
[1493]B.5.4.2.3 "Impad (PADder)"
Impad ensures that all the data token bodies include the word of 64 (or more than it). It is performed by padding the word of the last of a token with zero. Since it has 64 or more words in a body, a data token is not checked.
[1494]B.5.5 "Block execution"
B.5.5.1 "Hsppk"
Typically, splitting and packing are performed in one cycle.
[1495]B.5.5.1.1 "Splitting"
First, :IF (datatoken) as which a format is determined
IF (lastformat==1) use format 0a;
ELSE IF (run==0) use format 0;
ELSE use format 1;
And :format 0 format bit =0 as which a format bit is determined;
format 0a format bit = extension bit;
format 1 format bit = 1;
Since a code level must still be outputted when format 1 is used, new data should not be accepted in the following cycle.
B.5.5.1.2 "Packing"
It circulates through a packing procedure every four effective data input. The output of 16 bit word is formed from the effective word of the last currently held, and the word following it. An output is not effective, either, when this is not effective. The procedure is as follows. : [1496]

[Table 214]
x directs the bit of the undefined among front.
[1497]Between the effective cycles 0, since any words are not effective, it is not outputted.
[1498]An effective cycle number is maintained by the ring counter. The increment of it is carried out by the effective data and the accepted output from a splitter.
[1499]If a FLUSH (or picture end) token is received and the preparation whose token itself is an output is completed, a flash signal will be outputted to a DRAM interface and will reset an effective cycle to zero. If a flash plate token reaches things other than cycle 3, in order to guarantee that the token itself outputs, the flash signal must delay an effective cycle.
[1500]B.5.5.2 "Imodel"
B.5.5.2.1 "Imup (unpacker)"
Like the case of a packer, the last effective input is memorized, it is combined with the next input, and unpacking is made possible.
[1501]
[Table 215]
x directs the bit of the undefined among front.
[1502]An effective cycle number is maintained by the ring counter. Unpacked data contains the picture end decoded from the data of a token, a flash plate, and it. Besides, in addition, a format and an extension bit are decoded from unpacked data.
[1503]
formatbit is extn=(lastformat==1) 11 databody format=databody && (formatbit && lastformatbit)
Since [for token decoding] it is sent to imex.
[1504]All the data is deleted until a FLUSH (or picture end) token is unpacked, it is outputted to imex and block - and a signal are received from a DRAM interface (Valid is made low).
[1505]B.5.5.2.2 "Imex" (EXpander)
According to this invention, imex(es) are four State machines for expanding a run / level code outside. : and the State 0 whose State machine is as follows: Load a run count from a run code.
[1506]- State 1 : decrease a run count and output zero.
[1507]- State 2: -- input data and output level; -- the default State.
[1508]- State 3 : the illegal State.
[1509]B.5.5.2.3 "Impad (PADder)"
Impad is told about a data token header by imex. Next, it counts the number of the coefficients in a token body. Before becoming a coefficient of 64, after a token is completed, a zero coefficient is inserted by a token end and it is made to complete so that it may become 64 coefficients. For example, unextended data headers have a

zero coefficient of 64 inserted after them. The data token with 64 or more coefficients is not influenced by impad.
[1510]B.5.6 ″Register″
imodel and hsppk of this invention do not have a microprocessor register except for those SUNOPA.
[1511]
[Table 216]
The inside of front, a V= effective-bits;A= acceptance bit;E= extension bit;D= data bit.
[1512]B.5.7 ″Collation″
The selected stream flows through a Lsim simulation.
[1513]B.5.8 ″Test″
through token-buffers output SUNOPA, the test cover ridge to imodel in an input comes out, and there is, and it comes out through SUNOPA of the imodel itself in an output. Logic is covered with the scan chain of the imodel itself.
[1514]The output of hsppk can be accessed through Huffman output SUNOPA. Logic can be seen through the Huffman scan chain.
Section B.6 ″Buffer start-up″
B.6.1 ″Preface″
This section explains the method of buffer start-up and execution by this invention.
[1515]B.6.2 ″View″
In order to guarantee that a picture stream can be displayed smoothly and continuously, before being able to start decoding, a constant rate of data must be collected. This is called start-up conditions. A coding standard specifies the VBV delay which may be translated into data volume with required being collected mostly. It is the purpose of ″buffer start-up″ to ensure that all the streams fulfill the start-up condition before the data advances from token buffers and makes decoding possible. It is held by the notional gate (output gate) in a buffer in the output (getting it blocked reverse modeler) of token buffers. Once that start-up condition is fulfilled, this gate will only be opened for a stream.
[1516]B.6.3 ″Interface″
Bscntbit (buffer start-up bit counter) is in a data path, communicates with a 2-wire system interface, and is connected to a microprocessor. It branches to bsogl (buffer start-up output gate logic) with a 2-wire system interface. Bsogl controls imup (reverse modeler unpacker) which performs an output gate via a 2-wire system interface.
B.6.4 ″Block structure″
Bscntbit is in the data path between a start code detector and a coding data buffer. This one cycle block counts the effective data word which comes out of a block, and compares this number with the start-up conditions (or target) which are probably loaded from the microprocessor. bsogl will be told if a target is filled. Data is not

influenced by bscntbit. Bsogl is between bscntbit and imup (inside of a reverse modeler). As a matter of fact, it is a sequence of the indicator in which it is shown that the stream filled those targets. A sequence is moved by the stream (flash plate token which is got blocked and received in a data stream by imup) which comes out of a buffer when another "indicator" is accepted by imup. If a sequence is empty, the stream in imup will be stranded (if there is no stream in the buffer which was got blocked and filled the start-up target).
[1517]Although the sequence only has the limited depth, this can be infinitely expanded by breaking the sequence in bsogl, and a microprocessor enables it to supervise the sequence. The mechanism of these sequences is respectively called an internal sequence and an external sequence.
[1518]B.6.5 "Block execution"
B.6.5.1 "Bsbitcnt (buffer start-up bit counter)"
Bscntbit counts all the effective words inputted into buffer start-up. A counter (bsctr) is a programmable counter of 16 - 24 bit width. bsctr possesses a carry look-ahead circuit, in order to give sufficient speed for it. The width of Bsctr is programmed by ced bs prescale. it -- a bit -- 8-16 -- by making it high, it is carried out and, thereby, carry always comes to be sent. Therefore, those bits are used effectively. It is used for the comparison with a target (ced bs target) of top 8 bits of bsctr.
[1519]Comparison (ced bs count>=ced bs target) is performed by bscmp.
[1520]A target has a stream in the Huffman decoder, and when calculated by the microprocessor, it is pulled out from a stream. Therefore, it will only be set up after a stream start. Before start-up, targetvalid is set up low. The writing to ced bs target sets up target valid highly, and the comparison in bscmp is made to be performed. If it is shown that comparison is ced bs count>=ced bs target, target valid will be set up low. The target was filled.
[1521]A count will be reset if a target is filled. It is cautious of not resetting by a stream end. Besides, in addition, if it is before a stream end, counting will be made impossible after a target is filled. A count is saturated with 255.
[1522]If a stream end (getting it blocked flash plate) is detected in bsbitcnt, abs flush event will be made. After a stream is completed before a target is filled, an additional event (bs flush before target met event) is made. A block will be stranded if these events occur. The user can redo the target search of the following stream by this, In or the case of a bs flush before target met event event. It takes notice of that you make it follow the following stream until this which two target was not filled but was combined with the last stream reaches a target, or it writes in the target of the zero which will force it :1target met which is either of the following. The target for this following stream should be adjusted according to it.
[1523]B.6.5.2 "BSOGL (buffer start-up output gate logic)"
As mentioned above, Bsogl is an indicator sequence which directs that the stream

filled the target. A sequence type is set up by ced bs queue (an inside (0) or the exterior (1)). This is reset for choosing an internal sequence. The depth of a sequence determines the maximum number of the filled stream which can exist in a coding data buffer, Huffman, and token buffers. If this number is reached, bsogl will strand a data path in bsbitcnt (if it is got blocked and a sequence fills).
[1524]Use of an internal sequence does not need any operations from a microprocessor. However, it is if it is necessary to make the depth of a sequence increase (ced bs access in order to obtain access to ced bsqueue which should be set up), [set up and] target metevent and stream end event are made possible, and an external sequence can be set up by abandoning access.
[1525]An external sequence (count maintained by the microprocessor) is inserted in an internal sequence. an external sequence -- two event: -- it is maintained by target met event and stream end event. These are only respectively called servicequeue input, service queue output, and register ced bs enable nxt stream. As a matter of fact, target met event is an upstream end of the internal sequence which supplies a sequence. Similarly, ced bsenable nxt stream is a downstream end of an internal sequence which consumes a sequence. Similarly, stream end event is a request for supplying a downstream sequence, and;stream end event resets ced bs enable nxtstream. :/*TARGET MET EVENT*/ j= micro read (CED BS ENABLE NXT STM) for which two events should be served as follows : if(j==0)/*Is next. stream enabled?*/ (/*no, enable it*/ micro write (CED BS ENABLE NXT STM,1);
printf (″enable next stream **(queue =0x%x) n″, (context->queue));
}
else /*yes, increment the queue of ″target met″streams*/ {queue++;
printf (″stream already enabled **(queue=0x%x) n″, (context->queue));
}
/*STREAM EVENT*/ if (queue>0). /*are there any″target mets″left?*/ (/*yes, decrement thequeue and enable another stream*/queue--;
micro write (CED BS ENABLE NXT STM,1);
printf (″enable next stream **(queue =0x%x) n″, (context->queue));
}
else printf (″queue empty cannot enable nextstream ** (queue=0x%x) n″, queue);
micro write; (CED EVENT 1 and 1 << BS STREAM END EVENT) Although /*clear event* / sequence type can be changed into the exterior from an inside at any time (above-mentioned means), When an external sequence is empty (from the above-mentioned ″queue==0), In order to obtain access to ced bs queue which should be reset, It can change into an inside from the exterior by setting up ced bs access, carrying out the mask of target met event and stream endevent, and abandoning access.
[1526]On the other hand, the check of stream start-up conditions is made impossible,

ced bsqueue (exterior) is set up, the mask of targetmet event and stream end event is carried out, and ced bs enable nxtstream is set up. By this method, all the streams are always made possible.
B.6.6 "Microprocessor register"
[1527]
[Table 217]
[1528]
[Table 218]
-D is a register bit among front, -x is a register bit not existing, and -r is a reserved register bit -, in order to obtain access to the register of these, If it is not a case of an interrupt service routine, cedbs access is set as one, and it must be registered until it returns 1. Access is abandoned by setting ced bs access as zero.
[1529]Section B.7 "DRAM interface"
B.7.1 "View"
In this invention, a space decoder, a time decoder, and a video formatting part include a DRAM interface block respectively for the special chip. In all three devices, the functions of a DRAM interface are the data communications to the chip from external DRAM via the data communications from a chip to external DRAM, and the block address supplied by an address generator.
[1530]A DRAM interface is typically operated from an asynchronous clock to the clock of various blocks with which data is sent through it as opposed to an address generator. However, since a clock is operated the almost same cycle, this asynchrony can be processed easily.
[1531]Data is usually transmitted between a DRAM interface and the remaining chip within 64 bytes of block (the only exception is the prediction data in a time decoder). Transmission is generated with the device known as a "swing buffer." This is DRAM of the couple intrinsically operated in double buffered composition, and while a DRAM interface fills one RAM or makes it empty, another portion of a chip empties other RAM, or fills it. Another bus which carries an address from an address generator unites with each swing buffer.
[1532]Although each chip has four swing buffers, the function of these swing buffers changes with each cases. In the case of a space decoder, it is used in order that one swing buffer may transmit coded data to DRAM, It is used in order that another swing buffer may read coded data from DRAM, in order that the third swing buffer may transmit to DRAM the data by which tokenization was carried out, it is used, and it is used in order that the fourth swing buffer may read the data by which tokenization was carried out from DRAM. In a time decoder, in order that one swing buffer may write intra or the predicted picture data in DRAM, it is used, It is used in order that the second buffer may read intra or the predicted picture data from DRAM, and it is used in order that other two buffers may read prediction data to the front and back. In a

video formatting part, in order that one swing buffer may transmit data to DRAM, it is used, It is used in order that other three swing buffers may read data from DRAM, and it is because each is luminance (Y), red, and blue color difference data (Cr's reaching respectively Cb).
[1533]The following sections explain the operation of the DRAM interface by this invention, and it has one intrinsically same light swing buffer and one lead swing buffer as the operation of a space decoder DRAM interface. This is illustrated to the "DRAM interface" of Drawing 140.
[1534]B.7.2 "General DRAM interface"
In Drawing 140, all the interfaces to the block which supplies the interface to an address generator and data, and takes data are 2-wire system interfaces. The address generator can generate an address as a result of the receipt of a control token, or can only generate the fixed sequence of an address. A DRAM interface is a special method and processes the 2-wire system interface which unites with an address generator. Instead of holding an acceptance line highly, when ready for receiving an address, an address generator supplies an effective address, and processes the address, and a DRAM interface sets up the acceptance line between one clock period highly. In this way, a request / acknowledgement (REQ/ACK) protocol is performed.
[1535]The peculiar feature of a DRAM interface is the capability to connect an address generator and other blocks with the block which provides / accepts data separately thoroughly. For example, although the address generator can generate the address relevant to data in a light swing buffer, aren't any actions taken unless a light swing buffer signs that there is a data block whose preparation written in external DRAM was completed? However, any actions are not taken unless an address is supplied to the suitable bus from an address generator. Once one of the RAM in a light swing buffer is filled with data, before data input is stranded in other RAM (a 2-wire system interface accept signal is set up low), it will be filled thoroughly and "will be swung" by the DRAM interface.
[1536]The important thing which should be observed when you understand the operation of the DRAM interface of this invention, In the system constituted appropriately, it is quick at least to the same extent as a swing buffer and the sum total of all the average data speed during the remaining chip, and is that a DRAM interface can transmit data between a swing buffer and external DRAM.
[1537]Serving each DRAM interface for the next contains the method of determining which swing buffer it is. or [that this is generally "round-robin"] -- or it being either of the priority encoders and, In RAUNDOROBIN, the swing buffer served is a swing buffer which can be used next turn rarely turned recently, and, in the case of a priority encoder, some swing buffers have a priority higher than other buffers. The additional request will come from the refresh request generator which has a priority higher than other requests of all the in both cases. A refresh request is generated from a

programmable refresh counter via a microprocessor interface.
[1538]B.7.2.1 "Swing buffer"
Drawing 141 shows a light swing buffer. : whose operation is as follows -- one -- effective data is displayed in an input (data in). It is written in RAM1 and the increment of the address is carried out as each data pieces are accepted.
[1539]2) When RAM1 is full, input data gives up control, and in order to direct what the preparation from which RAM1 is read was able to carry out, send a signal to a lead side. This signal passes along between two asynchronous clock regimes, therefore passes three synchronous flip-flops.
[1540]3) The following data item which should reach an input side is written in RAM2 which is still empty.
[1541]4) If it directs that round-robin ****** is watch on which a priority encoder reads this swing buffer, a DRAM interface will read the contents of RAM1 and will write them in external DRAM. Then, like (2), a signal crosses an asynchronous interface, and is returned and what the preparation written in RAM1 once again was able to carry out is directed.
[1542]5) If it "is swung" before a DRAM interface empties RAM1 and an input side fills RAM2, As for the swing buffer, the accept signal will be low set up until it acts as the "swing back", since RAM1 is used by the input side, when data can be continuously accepted by a swing buffer, otherwise, RAM2 is filled.
[1543]6) This process is repeated infinitely.
[1544]Although the operation of a lead swing buffer is also the same, the data bus of an input and an output is reverse.
[1545]B.7.2.2 "Addressing of external DRAM and a swing buffer"
A DRAM interface is designed make available memory band width into the maximum. Therefore, it is arranged so that eight xeach 8 data block may be memorized by the same DRAM page. By this method, DRAM high-speed page access mode can be used thoroughly, one row address is supplied in that case, and many column addresses are supplied continuously. Besides, in addition, the data bus to external DRAM is made to 8, 16, or 32 bit width, and facilities for the amount of DRAMs used to enable it to suit the size of special application and the requirements for bandwidth are provided.
[1546](It is shown how the DRAM interface on a space decoder acts) In this example, an address generator provides a DRAM interface with a block address for [each] a lead swing buffer and a light swing buffer. This address is used as a row address for DRAM. 6 bits of a column address are supplied by the DRAM interface itself, and these bits are used also as an address further for swing buffer RAM. The data bus to a swing buffer is 32 bit width, and follows, When the bus width to external DRAM is narrower than 32 bits, before the following word is read from a light swing buffer, Or 2 to four external DRAM accesses (are related to a transmission direction [as opposed to external DRAM in a lead and a light]) before the following word is written in a lead

swing buffer must be performed.
[1547]In the case of a time decoder and a video formatting part, a situation is more complicated. In the following, these are explained independently.
[1548]B.7.3 "DRAM interface timing"
In this invention, a DRAM interface timing block uses a timing chain, in order to put the edge of a DRAM signal on one fourth correctly [a system clock cycle]. Two right-angled clocks from a phase-locked loop are used. These are together put in order to form notional 2x clock. Any one chain is made from two shift registers arranged in parallel on the opposite phase of 2x clock.
[1549]First, there is a phase for a page start cycle, and there is another phase for read/write/refresh cycle. It is programmable via a microprocessor interface in the length of each cycle, a page start chain has fixed length after that, and the length of a cycle chain changes suitably during a page start.
[1550]If reset, a chain will be cleared and a pulse will be made. This pulse moves along with a chain and is oriented by the state information from a DRAM interface. A DRAM interface clock is generated by this pulse. Each DRAM interface clock period corresponds to one cycle of DRAM. Thus, since it has the length from which a DRAM cycle differs, a DRAM interface clock is not constant speed.
[1551]A timing chain combines the pulse from the above-mentioned chain with the information from a DRAM interface, and generates an output strobe and enabling (notcas, notras, notwe, notoe).
[1552]Section B.8 "Inverse quantization device"
B.8.1 "Preface"
This writing explains the purpose of the inverse quantization device (iq) by this invention, operation, and execution.
[1553]B.8.2 "View"
An inverse quantization device reconstructs the coefficient from the quantized coefficient, quantization weight, and step size, and the all are transmitted within a data stream.
[1554]B.8.3 "Interface"
Iq is between a reverse modeler and reverse DCT in a data path, and is connected to a microprocessor. Data path connection is made via a 2-wire system interface. Input data is 10 bit width and output data is 11 bit width.
[1555]B.8.4 "Operation of an inverse quantization device"
B.8.4.1 "H261 formula"
[1556]
[Equation 1]
B.8.4.2 "JPEG type"
[1557]
[Equation 2]

B.8.4.3 ″MPEG type″
[1558]
[Equation 3]
B.8.4.4 ″JPEG variation type″
[1559]
[Equation 4]
B.8.4.5 ″-- others -- all the token″
All the tokens other than a data token must pass iq which is not quantized. :(a-1) <floor(a) <=a a>=0 to which :Floor (a) returns an integer as follows, and a<=floor(a) <(a+i) a<=0Qi are quantization coefficients there.
[1560]Ci is the reconstructed coefficient.
[1561]Wi and j are the values in a quantization table matrix.
[1562]i is the coefficient index which met zigzag.
[1563]j is a quantization table matrix number (0<=j<=3).
[1564]B.8.4.6 ″Multiple standard combined″
The above-mentioned standard and all those variations (by iq further.) The control data which must not be changed the even number of the conversion and all from - sign magnitude adding : and 1024 with the additional mailbox inverse quantization function :OUTPUT =(2 INPUT+k) (xy) / not more than 16 which can carry out a map to one formula to the complement expressions of 2. - result around [result] it goes to the nearest odd number toward zero is saturated in +2047 or -2048.
[1565]The function which the variable k, x, and y for each standard variation and they use is shown in Table 219 and Table 220.
[1566]B.8.4.6 ″Multiple standard combined″
[1567]
[Table 219]
[1568]
[Table 220]
B.8.5 ″Block structure″
The paragraph B.8.4.6 and Table 219, and Table 220 show that one structure can be used for multi-standard inverse quantization devices. : which shows the arithmetic block diagram in Drawing 142 ″an arithmetic block″ -- decoding of the token for loading the control for an arithmetic block to : and status register, or quantization table which can be functionally divided into two sections. - Decoding to the control signal of a status register.
[1569]A token is decoded in iqca which controls the bank of the following cycle, i.e., iqcb of a register. It controls access to four quantization tables in igram further. Arithmetic, i.e., two multipliers and post functions, is in iqarith. The perfect block diagram for iq is shown in Drawing 143.
[1570]B.8.6 ″Block execution″

B.8.6.1 ″Iqca″

In an invention, iqca is the State machine used in order to decode a token to the control signal for registers in igram and iqcb. It is better to explain the State machine as a State machine for each tokens, since it is reset by each new token. for example, : -- QUANT SCALE (B. refer to 8.7.4 and QUANT SCALE) and QUANT TABLE (B. refer to 8.7.6 QUANT TABLE) -- : which is as follows [code / of business] -- if (tokenheader==QUANT SCALE)

```
{sprintf(preport, ″QUANT SCALE″);
reg addr=ADDR IQ QUANT SCALE;
rnotw=WRITE;
enable=1;
}
if(tokenheader==QUANT TABLE)
/*QUANT TABLE token*/switch(substate)
{case 0: /*quantisation table header*/sprintf(preport, ″QUANT TABLE %s s0″, (headerextn ? ″(full)″ : ″(empty)″));
nextsubstate=1;
insertnext=(headerextn ? 0:1);
reg addr=ADDR IQ COMPONENT;
rnotw=WRITE;
enable=1;
break;
case 1: /*quantisation table body*/ sprintf(preport, ″QUANT TABLE %s s1″, (headerextn ? ″(full)″ : ″(empty)″));
nextsubstate=1;
insertnext=(headerextn ? 0 : (qtm addr 63==0));
reg addr=USE QTM;
rnotw=(headerextn ?WRITE : READ);
enable=1;
break;
default: sprintf (preport, ERROR in iq quantisation table tokendecoder **(substate %x) n, substate);
break;
}
}
```

When a substate is the State in a token, QUANT SCALE has only one substate, for example. However, QUANT TABLE has two substates, one is a header, and other one is a token body.

[1571]The State machine is mounted as a PLA. What kind of word line does not make it generate, either, and an unrecognized token does not make default (it is harmless)

control output also to PLA.
[1572]Therefore, iqca supplies an address to igram from a BodyWord counter, for example, inserts a word in a stream in un-extending QUANT TABLE (B. refer to 8.7.4). While this keeps an output effective, it is attained by stranding an input. In the continuing block (iqcb or iqarith), a word can be fulfilled by right data.
[1573]iqca is 1 cycle in the data path controlled by a 2-wire system interface.
[1574]B.8.6.2 "iqcb"
In an invention, iqcb holds iq status register. iqcb loads or unloads these from a data path to / under control of iqca.
[1575]A status register is decoded by the control wire for iqarith in order to control XY multiplier term and a post quantization function (see Table 219 and Table 220).
[1576]It dissociates here and the sign binary digit of a data path is sent to a post quantization function. The zero value word on a data path is detected here. Arithmetic is disregarded after that and it acts to a data path as Max of the zero. "zero yne this [whose] is iq; it is the easiest method of following zero out" spec.
[1577]In the status register, only when the register iq access is set as one and 1 is returned, it is accessible from a microprocessor. In this situation, iqcb stops a data path, and has the value which was stabilized in this way as for the register, and any data is not converted in a data path.
[1578]Iqcb is 1 cycle in the data path controlled by a 2-wire system interface.
[1579]B.8.6.3 "Iqram"
Each must support Iqram to four quantization table matrices (QTM) which are 648 bits. Therefore, it is 2568-bit 6 transistor RAM in which one lead or one light is possible for every cycle. RAM is surrounded by the 2-wire system interface logic which receives the control, and writes in data from iqca. It reads data to iqarith. Similarly, igram occupies the cycle in the same data path as iqcb.
[1580]When iq access reads one over, RAM is read from a microprocessor and written in. RAM is put behind a keyhole register and iq qtm keyhole, and an address is carried out by iq qtm keyhole addr. iq Access to qtm keyhole will carry out the increment of the address which is held in iq qtm keyhole addr and which it points out. Similarly, iq qtm keyhole addr can be written in directly.
[1581]B.8.6.4 "iqarith"
It is cautious of iqarith continuing for three cycles, being connected on a pipeline, and having three functions by which a split is carried out. A function is discussed in the following (see Drawing 140).
[1582]B.8.6.4.1 "XY multiplier"
This is carry save the unsigned multiplier of the 5(X)x8(Y) bit supplied to a data path multiplier. A multiplier and a multiplicand are chosen with the control wire from iqcb. Multiplication is in the 1st cycle and a decomposition adder is in the 2nd cycle.
[1583]In the input to a multiplier, in order to read QUANT TABLE to a data path, it

may act to a data path as Max of the data from iqram.
[1584]B.8.6.4.2 ″(XY) * data path multiplier″
The split of the carry save unsigned multiplier of this 13(XY) x12 (data path) bit is continued and carried out to three cycles of a block. It is seven and two of the remainder [cycle / 3rd] in three partial products and the 2nd cycle at the 1st cycle.
[1585]Since all the outputs from a multiplier are 2047 (non coefficient) or less or are saturated in +2047/−2048, it does not need to be decomposed by top 12 bits. Therefore, a decomposition adder is only 2 bit width. Zero detection is enough as a SACHU ration signal about the remainder of the bit of a high order.
[1586]B.8.6.4.3 ″Post quantization function″
− which changes a post quantization function into a two's complement display from − sign magnitude adding −1024 − output which saturates − result which goes around [all even number] to the nearest odd number toward zero in +2047 or −2048 is set as zero (B. refer to 8.6.2).
[1587]The first three functions are performed by a 12-bit adder (tied to the 2nd and the 3rd cycle on a pipeline). From this, it realizes what what each function needs is, and these are combined on one adder.
[1588]
[Table 221]
It will recognize, if it is a person skilled in the art, but since it depends for these functions mutually when combined, it must be careful when carrying out re-PURUGU ramming of these.
[1589]It acts to a data path as Max of SACHU ration value, zero, and the zero+1024 in the end of the 3rd cycle.
[1590]B.8.7 ″Inverse quantization device token″
The following explanation defines the act of an inverse quantization device for each token tp which an inverse quantization device answers. In all the cases, a token is sent at the output of an inverse quantization device. In almost all cases, a token is not corrected by the inverse quantization device except for the exception described below. All the unrecognized tokens are sent to the output of an inverse quantization device with un-correcting.
[1591]B.8.7.1 ″Sequence start″
As for this token, the registers iq prediction mode [1:0] and iq mpeg indirection [1:0] are set as zero.
[1592]B.8.7.2 ″Coding standard″
Based on the standard (MPEG, JPEG, or H.261) of the present when this token is decoded, the suitable value for iq standard [1:0] is loaded.
[1593]B.8.7.3 ″Prediction mode″
This token loads iq prediction mode [1:0]. Although a prediction mode token owns two or more bits, an inverse quantization device only needs access to the bit of the two

minimum orders. These judge whether the Intra coding of the block is carried out.
[1594]B.8.7.4 ″Quantizing scale″
This token loads iq quant scale [4:0].
[1595]B.8.7.5 In day t this invention, this token owns a actual quantization coefficient. These are loaded to iq component [1:0] including two bits from which a token head discriminates a color component. The following token word of 64 contains a quantization coefficient. These are corrected as a result of an inverse quantization process, and are replaced with the reconstructed coefficient.
[1596]When the extended word of 64 does not exist in a token correctly, the act of an inverse quantization device is not limited.
[1597]The data token in the input of an inverse quantization device owns a quantization coefficient. These are displayed at 11 bits (10 bit + sign binary digit) in a sign magnitude format. Although the value ″minus zero″ should not be used, it is correctly interpreted as zero.
[1598]The data token in the input of an inverse quantization device owns the reconstructed coefficient. These are displayed at 12 bits (11 bit + sign binary digit) in a two's complement format. Probably, the data token in an input has same number as having had in the input of an inverse quantization device of token extension words.
[1599]B.8.7.6 ″Quantization table (QUANT TABLE)″
This token can be used since a new quantization table is loaded, or in order to read the present table. In the inverse quantization device, typically, the token will be used, since the new table decoded from the bit stream is loaded. In the quantizer ahead of an encoder, the operation which reads the present table is useful, when the table will be encoded in a bit stream.
[1600]A token head contains two bits which specify the table numbers which will be used. These are placed into iq component [1:0]. It is cautious of now this register containing not a color component but ″table numbers.″
[1601]When the extension bit of a token head is 1, it is expected that an inverse quantization device will have an extended token word of 64 there correctly. Each is interpreted as quantization table value and placed into the location which the suitable table which starts with location zero follows. The 9th bit of each extended token word is disregarded. A token is the further usual method and is sent also to the output of an inverse quantization device with un-correcting.
[1602]The inverse quantization device will read the location which the suitable table which starts with location zero follows, when the extension bit of a token head is zero. Each location becomes an extended token word (the 9th bit becomes zero). Probably, the token includes the extended token word of 64 correctly in the end of this operation.
[1603]The operation of the inverse quantization device which replies to this token is limited for [no] zero and the extended word numbers except 64.

[1604]B.8.7.7 "JPEG table selection"
To the table numbers from/to iq jpeg indirection, this token is used in order [which loads translation of a color component] to do /unloading of. These translations are used in JPEG and other standards.
[1605]A token head contains two bits which specify the color component which is interested now. These are put on iq component [1:0].
[1606]When the extension bit of a token head is 1, a token contains lowest 2 bits written in an one extended word and iq jpeg indirection[2iq component[1:0]+1:2iq component [1:0]] location. Value to the extent that it was read just now becomes a token extension word (top 7 bits becomes zero). Probably, at the time of the end of this operation, the token includes one token extension word correctly.
[1607]
[Table 222]
B.8.7.8 "MPEG table selection"
This token is used in order to define whether the quantization table which a default or a user limits between processing through an MPEG standard is used. A token head contains 2 bits. It is determined in which bit the bit 0 of a header is written when iq mpeg indirection is written in. The bit 1 is written in the location.
[1608]iq Probably, a mpeg indirection [1:0] register only needs to use this token, when a user limited quantization table is sent in a bit stream, since it is cleared by the sequence start token.
[1609]B.8.8 "Microprocessor register"
B.8.8.1 "iq access"
In order to obtain micro-processor access to one of iq registers, iqaccess must be set as one, and it must be registered until it reads 1 over (B. refer to 8.6.2). If it neglects carrying out this, the register currently read will be in the state where it is still controlled by the data path, and will not be stabilized. In igram, it is locked out and access returns zero.
[1610]iq When zero is written in access, control will be parted with to a data path.
[1611]B.8.8.2 "Iq coding standard [1:0]"
This register holds the coding standard performed by the inverse quantization device.
[1612]
[Table 223]
This register is loaded by coding standard value.
[1613]Although this is two bit registers, 8 bits is assigned to the present memory map and the thing more than the above-mentioned standard can be processed in future mounting.
[1614]B.8.8.3 "Iq mpeg indirection [1:0]"
These two bit registers are used in order to record which quantization table is due to be used between MPEG decoding operations.

[1615]Iq mpeg indirection [0] controls the table used for the Intra coding block. If it is 0, the quantization table 0 is used and including a default quantization table is expected. If it is 1, the quantization table 2 is used and including a user limited quantization table is expected.
[1616]This register is loaded by the MPEG TABLE SELECT token, and is set as 0 by the sequence start token.
[1617]B.8.8.4 ″Iq jpeg indirection [7:0]″
It is determined which four quantization tables these eight bit registers use for [each] four possible color components generated in a JPEG scan.
[1618]- A bit [1:0] holds the table numbers used for the ingredient 0.
[1619]- A bit [3:2] holds the table numbers used for the ingredient 1.
[1620]- A bit [5:4] holds the table numbers used for the ingredient 2.
[1621]- A bit [7:6] holds the table numbers used for the ingredient 3.
[1622]This register is influenced by the JPEG TABLE SELECT token.
[1623]B.8.8.5 ″iq quant scale [4:0]″
This register holds the present value of a quantizing scale function. This register is loaded by the QUANT SCALE token.
[1624]B.8.8.6 ″iq component [1:0]″
This register holds the value translated into a quantization table matrix (QTM) number. It is loaded by the token number.
[1625]The color component of the block with which the data token header is going to be processed by this register is loaded. In order that this information may determine a QTM number, it is used only in JPEG and a JPEG variation, and it is processed by reference to iq jpeg indirection [7:0]. iq component [1:0] is disregarded in other standards. A JPEG table selection token makes a color component load to this register. Then, it is used as an index to iq jpeg indirection [7:0] accessed by the token body.
[1626]A quantizing scale token makes a QTM number load to this register. This table is read from a table, in order to form after that the token which was loaded from the token or was extended suitably (when the extended form of a token is used).
[1627]B.8.8.7 ″iq prediction mode [1:0]″
These two bit registers hold the prediction mode used for the block following it. the only directions for which an inverse quantization device uses this information -- intra -- it is determining whether coding is used or not. If both the bits of a register are 0, the Intra coding of the following block is carried out.
[1628]This register is loaded by the prediction mode token. This register is set as 0 by the sequence start token.
[1629]Iq prediction mode [1:0] does not have influence of what on the operation in JPEG and JPEG variation mode, either.
[1630]B.8.8.8 ″Iq jpeg indirection [7:0]″

Iq jpeg indirection is used as a lookup table for translating a color component into a QTM number. Therefore, iq component is used as an index to iq jpeg indirection, as shown in Table 222.
[1631]This register location is directly written in by the JPEG table selection token, when the extended form of a token is used.
[1632]When the non-extending form of a token is used, reading appearance of this register location is directly carried out by the JPEG table selection token.
[1633]B.8.8.9 ″Iq quant table [3:0], [63:0], [7:0]″
There are four quantization tables and each has a location of 64. Each location is 8-bit value. The value 0 should not be used in any locations.
[1634]These registers are mounted as RAM explained in Igram of B.8.6.3.
[1635]These tables can be loaded using a quantization table token.
[1636]It is cautious of the data in these tables being memorized to a zigzag scan order. Many documents show quantization table value as a rectangular number array of 8x8. Usually, it is at least in the upper left, level frequency increases from the left to the right, and vertical frequency increases DC term downward from a top. This **** table must be read along a zigzag scan pass as it is placed into a quantization table with i which a number follows.
[1637]B.8.9 ″Microprocessor register map″
[1638]
[Table 224]
B.8.10 ″Examination″
through output SUNOPA of an inverse quantization device, the test cover ridge to the inverse quantization device in an input comes out, and there is, and it comes out through SUNOPA of the inverse quantization device itself in an output. Logic is covered by the scan chain of the inverse quantization device itself.
[1639]Authorization of a ramtest signal will obtain access to igram regardless of iq access.
Section B.9 ″IDCT″
B.9.1 ″Preface″
The purpose of explaining a reverse discrete cosine transform (IDCT) block here is to provide the engineering sources of information for IDCT. It includes the following information.
- The purpose of IDCT, and the main feature and it were designed how, and it proved, or - structure and also its purpose are to provide sufficient information for a person skilled in the art, in order for such explanation to do the following work easy or to help.
[1640]- Recognition of IDCT as a ″silicon macro function″, correction of development and IDCT of the test program for integration and IDCT silicon of IDCT to another device, redesign, or development B.9.2 of maintenance and forward direction DCT blocks ″View″

A discrete cosine transform/ZIG ZAG (DCT/ZZ) changes by being on the block of a pixel, and each block displays screen area with an 8 pixel [in height] x width of 8 pixels. The purpose of conversion is to display a pixel block in the frequency domain classified according to frequency. Although eyes are sensitive to the DC component in a picture, it is made, as for frequency data, for each ingredient to decrease magnitude separately according to the sensitivity of eyes to a high frequency component, since it is not so sensitive. The magnitude reduction process is known as quantization. A quantization process decreases the information included in a picture, that is, a quantization process is loss nature. A loss nature process performs an overall data compression by deleting information of a certain kind. Frequency data is classified so that high frequency may make it be easy to be quantized by 0, and all appear continuously. Continuous 0 means that coding the data quantized by using a run length coding plan produces the further data compression, although run length coding is not generally a process of loss nature.
[1641]The frequency data according to which an IDCT block (this actually contains reverse ZIG ZAG RAM or IZZ and IDCT) is classified is taken, and it is changed into spatial data. This reverse classification process is a function of IZZ.
[1642]The picture pressure reduction system with which an IDCT block forms the part specifies a pixel as an integer. It means that an IDCT block must take an integral value and this must produce an integral value. However, since the function of IDCT is not based on an integer, an internal number display uses a fraction portion in order to maintain internal accuracy. Although perfect floating point calculation is preferred, fixed point calculation is used in mounting indicated here. When fixed point calculation is used, there are a little losses of accuracy, but the accuracy about this mounting is over the accuracy specified by H.261 and IEEE.
[1643]B.9.3 "Design target"
According to this invention, the main design targets were things which use the minimum silicon area and for which an exact IDCT block is designed functionally. Although running a design with the clock rate of 30 MHz under the specified operating condition was called for, it was thought that it should also have the applicability over the future. Probably the higher rate of a clock is needed in the future, and the structure of a design makes this possible in a possible place.
[1644]B.9.4 "Explanation of an IDCT interface"
The IDCT block has the following interfaces.
[1645]- The token data output port microprocessor interface port system service input port test interface and resynchronization signal of the token data input port bit width of 12 bit width
Both the token dataport is a 2-wire system interface type of the standard mentioned above. The illustrated width has pointed out not the total number of wires of a port but the number of bits in data display. Besides, in addition, the clock signal and reset

signal which are used for the resynchronization to the output of a pre- block unite to an input token dataport. It unites to an output token dataport and there are two resynchronization clocks used by the block following the next.
[1646]A microprocessor interface is a standard and uses a 4-bit address. There are three selection inputs decoded externally, and they are used in order to choose an event, an internal register, and the address space for test registers. This mechanism provides the flexibility for carrying out the map of the IDCT address space to a different position in a different chip. There are one event output, idctevent, two i/o signals and n derrd, and nserrd, and they are event tri-state data wires externally connected to the suitable bit of IDCT and the non-data bus of a microprocessor.
[1647]The service port of a system comprises a standard clock signal, a reset input signal, a 2 phase override clock, and a related clock override mode selection input.
[1648]A test interface comprises a JTAG clock signal and a reset signal, scanning path data, a control signal and ramtest, and a chiptest input.
[1649]Since access of a microprocessor is not needed in the usual operation in order that IDCT may achieve the specified function, a microprocessor port is inertness. Similarly, a test interface is activity only when a test or a check is required.
[1650]B.9.5 "Operation base for discrete cosine transforms"
In video bandwidth range compression, input data displays the rectangular portion of a picture. Therefore, the conversion applied must be two dimensions. Although it is difficult to calculate two-dimensional conversion effectively, two-dimensional DCT has a separable property. Disengageable conversion may be calculated in accordance with each dimension apart from other dimensions. using the one-dimensional IDCT algorithm designed especially since it maps to hardware, in order to perform this --; -- the algorithm is not suitable for a software model. A one-dimensional algorithm is continuously applied, in order to obtain a two-dimensional result.
[1651]The operation definition of two-dimensional DCT for the pixel blocks of NxN is as follows. : [1652]
[Equation 5]
The above-mentioned definition is equal to performing compatibility with a matrix and imposing two NxN matrices twice continuously between multiplication, in operation. One-dimensional DCT is equal to imposing two NN matrices in operation. In operation, in a two-dimensional case, it is among a :Y=[XC] TC type, and C is a matrix of the paragraph of cosine.
[1653]Thus, occasionally DCT may be explained by the paragraph of matrix operation. Although explanation of a matrix is convenient because of operation reduction of a fraction to the lowest terms of conversion, it must emphasize that this only simplifies a notation. It is cautious of the paragraph of 2/N governing DC levels. Constant c (j) and c (k) are known as a normalized function.
[1654]B.9.6 "IDCT conversion algorithm"

The algorithm used in order to calculate actual IDCT conversion so that it may explain in detail after the following should be a ″high-speed″ algorithm. The algorithm used is optimized for efficient hardware structure and mounting. The main features of an algorithm are conversion of the algorithm designed since big symmetry is produced by between use of =2 scaling for removing one multiplication, a higher rank, and low rank sections. This symmetry can be made to carry out the reuse of many of most expensive operation part efficiently.
[1655]In the diagram (Drawing 145) showing an algorithm, the symmetry between a higher rank half and a lower half is clear at an omitted portion. The column of the last of a subtractor also has antimere with the adder, and an adder and a subtractor can be comparatively combined with a low price (four an adder / subtractors become quite smaller than four adders and four subtractors which are illustrated).
[1656]It notes that all the outputs of one-dimensional conversion are estimated by =2. This means that the last two-dimensional answer is estimated by 2. In the rounding stage by final SACHU ration and shifting, this can be corrected easily.
[1657]The illustrated algorithm was coded in the double precision floating point C, and this result was compared with the standard (using clear matrix multiplication) IDCT. It was used in order that the following stage might code the bit exact integer version of the algorithm in C (timing information is not included), and probably, it has been used in order to check the performance and accuracy of an algorithm, when the algorithm was probably mounted on silicon. The mistake in permissible conversion is specified in the H.261 standard, and this method is used in order to perform a bit exact model and to measure the accuracy told.
[1658]Drawing 146 shows the method of illustrating the similarity between a higher rank section and a low rank section, an overall IDCT structure is shown and also the point with required memorizing an intermediate result is shown. In order to calculate separately a higher rank section and a low rank section, time multiplexing of the circuit is carried out.
[1659]B.9.7 ″IDCT conversion structure″
As mentioned above, an IDCT algorithm is optimized for an efficient structure. It is a fixed coefficient rather rather than the main features of the structure produced as a result have multiple-purpose multiplier of :, and the expensive, important reuse and small number of math operation and all that are as follows (multiplier size is decreased and the necessity for another coefficient memory is removed).
- A small number of latch, So that only one decomposition operation may be required for every - pipeline stage slightly required in order to connect structure on a pipeline.
- by which operation is arranged - which can be arranged so that a result may be produced to a natural order -- there is not complicated crossbar switching or important multiplexing (in mounting with final both, it is expensive)
- In order to remove two carry save operations (addition and one subtract one), . The

advantage and each stage pulled out from a decomposition result take four clock cycles. That is, by only changing decomposition operation into a large and high-speed carry look-ahead version from the structure which makes it possible to remove the necessity for very high-speed (it is big) math operation, and a small and late influencing carry, Structure which will support operation far more nearly high-speed than the present 30-MHz pixel clock operation. Since decomposition operation needs the maximum portion of required time in each stage, while speeding up only such operations has the serious influence for overall operation speed, the increase in the overall size of conversion is comparatively small, and ends. The increase in speed can be attained also by making the depth of pipelining increase.
[1660]- Control of conversion data flow is dramatically clear, and efficient.
[1661]The map of the algorithm is carried out to the hardware resources of a small lot, and the diagram (Drawing 150) of the one-dimensional conversion fine structure shows the method by which required performance is obtained. Control of this structure is attained by fitting a "control shift register" to a data flow pipeline. This control is clear to a design and efficient in a silicon layout.
[1662]The control signals (latch, sel byp, etc.) nominated on Drawing 150 are various enable signals used in order to control a latch.
A signal flows.
The clock signal to a latch is not illustrated.
[1663]While conversion structure makes conversion size the minimum, the details about some mounting become important from the point of enabling it to meet the required accuracy standard. The art generally used is mainly divided into two classes.
[1664]- Maintenance of the maximum dynamic range which has fixed word width by each intermediate state by controlling a fixed-point position separately.
[1665]- (the increase in accuracy by making the word width of the whole conversion only increase -- not but) in order for the selection operation of operation operation to attain accuracy -- use of a statistical definition of the requirements for accuracy.
[1666]Probably, the clear method of designing conversion comprised only performing a fixed point with the fixed word width made into sufficient size, in order to attain accuracy. This approach makes an unfortunate thing produce consequential more big word width, therefore makes it produce big conversion. The approach used in this invention is the method of carrying out the maximum use of the dynamic range which can be used for a special mean value, enables it to change a fixed-point position through conversion, and attains the accuracy in which the maximum is possible.
[1667]Since the result which can be used is specified statistically, in order to improve overall accuracy, alternative adjustment may be performed to every mean value. The adjustment chosen is mere operation of LSB calculation, and it hardly requires expense. As an alternative of this art, although word width may be increased, remarkable expense starts. Adjustment is giving weightings in the predetermined

direction to the result effectively, if the tendency for a final result to go to a predetermined direction and counter direction before is found out. By adjusting the fragmentary portion of a result, the overall average of these results can be shifted effectively.

[1668]B.9.8 "IDCT block line description of drawing"

The block diagram of IDCT shows all the blocks relevant to processing of a token stream. This diagram and Drawing 147 do not show crocking, a test, micro-processor access, and an event mechanism in detail. The snow pub rock used in order to provide test access is not illustrated in the diagram.

[1669]B.9.8.1 "Data error checker"

The first blocks are a data error checker and a collector, they choose the token stream of 12 bit width, /Make it, analyze this stream, and are called decheck which checks a data token. Other tokens of all the are disregarded and are sent directly. The check carried out is performed for a data token with the number of extension which is not equal to 64. It deficient(s) (insufficient) (<64 extension), and a possible error is, and is called dct too few event and supernumerary (superfluous) (>64 extension), and idct too many event. Although this **** error is signed by a standard event mechanism, a block also tries easy error recovery by operation of a token stream. In the case of an insufficient error, "0" value extension is packed (input acceptance is stopped and insertion is performed), and a data token completes 64 exact extension. In the case of a superfluous error, as for an extension bit, the 64th extension is forced into "0", and all the excessive extension is removed from a token stream.

[1670]B.9.8.2 "Reverse ZIG ZAG"

/The blocks next to a space decoder are reverse zigzag RAM and izz, and this also chooses the token stream of 12 bit width, and makes them. Like other the blocks of all the, although a stream is analyzed, only a data token is recognized. Other tokens of all the are sent without being changed. An extended order is changed although a data token is also sent. This block considers an exact data token (getting it blocked only 64 extension) as reliance. Operation will not be specified if this is not true. Rearrangement is performed according to a standard reverse zigzag pattern, and it is carried out by default in order to provide the data horizontally scanned in an IDCT output. Since the output scanned vertically is provided, it is also possible to change ordering. In addition to standard IZZ ordering, this block performs excessive rearrangement of an 8-word each low. This is performed on account of the special requirements for IDCT one-dimensional transformation blocks, and produces the low outputted not to the order (0, 1, 2, 3, 4, 5, 6, 7) but to the order (1, 3, 5, 7, 0, 2, 4, 6).

[1671]B.9.8.3 "Input formatting part"

The following blocks are an input formatting part and ip fmt, and it forms data input for the 1st dimension of IDCT conversion. This block has a token stream input of 12 bit width, and a token stream output of 22 bit width. In order that a data token may move an integral part to the right validity in an IDCT conversion standard 22 bit width word, it is shifted to the left and a fraction portion is set as 0. This means that there is a 10-bit fraction in this point. None of other tokens is shifted but an excessive intact bit is only set as 0.

[1672]B.9.8.4 The "one-dimensional conversion - first origin"

The following blocks currently illustrated are the first one-dimensional IDCT conversion block and oned. This inputs / outputs the token stream of 22 bit width, a stream is analyzed and a data token is recognized so that it may be usual. Other tokens of all the are sent with no changing. A data token passes the data path which achieves one-dimensional execution of the reverse discrete cosine transform of 8x8 and which was connected on the pipeline. There is a 7-bit fraction in the output of the 1st-dimensional one in data word. Other tokens of all the only suit data conversion waiting time, and pass the mere shift register data path rearranged by the token stream before an output.

[1673]B.9.8.5 "Transposition RAM"

In the way it processes a token stream, the transposition RAMtram resembles reverse zigzag RAM in many respects. Although the width (22 bits) of the token processed differs from the rearrangement carried out, in respect of others, it acts similarly, and many of the control logic is actually shared. Here, a low is rearranged additionally because of the requirements for the following IDCT dimension with fundamental swapping to the low of a column.

[1674]B.9.8.6 One-dimensional conversion - the "2nd dimension"

The following block currently illustrated is another example of a one-dimensional IDCT conversion block, and is the same as the 1st dimension in all respects. There is a 4-bit fraction in the output of this dimension.

B.9.8.7 "Round and SACHU ration"

Although a round SACHU ration block and ras choose the token stream of 22 bit width including data extension of a 22-bit fixed point format and the token stream of 9 bit width is outputted, In that case, it went around [data extension] for the integer (going to +ve infinity), and is saturated in a 9-bit two's complement display, and other tokens of all the are sent directly.

[1675]B.9.9 "Hardware description of a block"

B.9.9.1 "Standard block structure"

There is a notional structure of a standard as shown in Drawing 148 for all the blocks which process a token stream. This separates a 2-wire system interface latch from the section which carries out operation of a token stream. Excessive internal blocks (RAM core etc.) are contained in the variation on this structure. In some illustrated blocks, for the requirements of carrying out the grouping of the all the "data path" logic together, the structure is not made not much clear in a schematic diagram (in spite of still existing actually), but separates this from the cell logic of all the standards. In the very easy block of ras etc., latched outaccept can be put on a direct entry 2-wire system latch, without carrying out a logical operation.

[1676]B.9.9.2 ″Decheck-data error checking / recovery″

The block of the beginning in a token stream is corrected with data checking, as clearly written with the block diagram view section. The error detected is processed by a standard event mechanism, the mask of it can be carried out to an event, and the block can stop according to event mask status continuing with a recovery procedure, when an error is detected. Since IDCT should not look at the data token which is not right, the recovery which it tries becomes [whether what may be a serious problem try to be included and], and is only an easy trial.

[1677]This block has the depth of the pipeline of two stages, and is thoroughly performed in zcells. An input 2-wire system interface latch is a ″front″ type thing, and when it is after another power supply organization, it means that all the inputs arrive at a transistor gate, and it can be made to perform safe operation as what has this (it is before IDCT) block before that. This block analyzes a token stream and acts by sending a non-data token directly. If a data token is discovered, a count will be started from the number of extension found after the header. When a count is not 63 and it is found out that an extension bit is ″0″, it is emitted by the error signal (it is going to event logic), and according to the state of the mask bit for the events, decheck stops -- having (it is got blocked, and an input is not accepted any longer, or an output is not emitted) -- or error recovery is started. A counter is used for the recovery mechanism for deficient errors in order to control insertion to the token stream of the number of right extension (the value inserted is always ″0″). Clearly, although an input is not accepted, this insertion is continued by one side. A supernumerary error will be emitted, if the extension bit of the 64th extension is not ″0″ and it will be found out, The word which continues after all with the extension bit which the data token was completed by forcing ″0″ into an extension bit, and was set as ″1″ is deleted from a token stream by outputting invalid, although it continues accepting data.

[1678]Two error signals are not permanent (unless a block is stopped), that is, only an error signal remains from the point from which an error is detected with activity until recovery is completed. This is one perfect cycle at its minimum, and, in the case of a superfluous data token, is continued eternally infinitely.

[1679]B.9.9.3 ″Rearrangement of Izz and tram-RAM″

Since izz (reverse zigzag RAM) and tram (transposition RAM) have many similarity from the carry difference, they take the variation of the same function into consideration together here. These blocks choose a token stream, and while they send other tokens of all the with no changing, they rearrange extension of a data token. Although the extended width processed differs from the sequence of rearrangement, the big section of the control logic for each RAM is the same, and is systematized by the ″common control″ block actually illustrated in the schematic

diagram for each RAM. Since the difference in width does not have influence of what on this control section, either, it only needs to use a "sequence address generator" which is different in each RAM with a RAM core and the suitable 2-wire system interface block of width.
[1680]The overall act of each RAM is intrinsically the same as the thing of FIFO. This is true in a token level and special correction to an output order is made for the extended word of a data token. The depth of FIFO is 128 stages. Since the output of back FIFO from which the output start of the data token was detected is supported, this needs to satisfy the requirements for 30 MHz maintainable through a system. This is because it is requiring that perfect 64 extended blocks should be brought together in FIFO, before the feature of the rearrangement sequence used can begin a rearrangement output. If it says more correctly, reverse ZIG ZAG differs from the required minimum number for transposition sequences, and when it is both, it is less than 64 for how many minutes. However, the complexity of controlling FIFO with the length which is not BEKI of 2 means that small saving in a RAM core will probably be more important than the additional complexity of required control logic. A RAM core is performed by the design which can be made to do a lead and a light (to the same address or another address) in one 30-MHz cycle. This means that RAM operates it effectively by the cycle time which is 60 MHz of insides. The ranges of rearrangement operation are 0-63, however it is carried out by generating not a natural order but the special sequence of a read address (sequenceaddress generation). A required sequence is specified by the standard zigzag sequence (for eight horizontal or vertical scanning), or a sequence required for the usual matrix transposition. These standard sequences are further rearranged after that by the requirements (that is, not (0, 1, 2, 3, 4, 5, 6, 7) but (1, 3, 5, 7, 0, 2, 4, 6)) for outputting each low on account of the requirements for an IDCT conversion one-dimensional block within odd / even number format.
[1681]Transposition address sequence generating is completely clear in algorithm. Direct transposition sequence generating only needs a low and separate generating of a column address, and both they are performed at a counter. The requirements for low rearrangement mean only being generated not with a counter with a natural row address but with the rather easy special State machine.
[1682]Since it generates in algorithm, a reverse zigzag sequence is not so clear. In order to hold all the 64-bit value of this fact, therefore an address, small ROM is used, and in order to change between vertical scanning modes with it being level, the address of this is carried out at the counter of the low and column which may be swapped. The generator of a ROM base is designed very quickly and performing forward direction ZIG ZAG (ROM sand-casting), or adding other alternative sequences in the future also has the advantage that they are trifles.
[1683]B.9.9.4 "One "Oned"--dimensional IDCT conversion"

This block has the pipeline depth of 20 stages, and if a pipeline damps, he will become hard. Since this rigidity simplifies a design greatly and a pipeline's depth is not so large, overall power should not be affected too much, but RAM which provides a constant rate of bufferings still lingers, and both dimensions are inherited.
[1684]Although a block follows standard structure, it has another path internally for all the items of the others which should be sent with data token (process should be carried out) extension and no changing. It is cautious of the schematic diagram being drawn by the special method. first, the requirements of carrying out the group division of all the data path logic together -- therefore -- next, the requirements (this explains control logic by a top level) of enabling code generating edited automatically -- therefore .
[1685]A token is respectively sent through two different parallel paths, before a usual passage is analyzed and the value of data extension and others is rearranged by the multiplexer in front of an output 2-wire system interface latch block. Since value cannot be sent through a conversion data path with no changing, a parallel path is required. The waiting time of a conversion data path suits with an easy shift register, in order to process the remainder of a token stream.
[1686]The control section of oned needs to analyze a token stream and needs to control splitting of a token, and recombination. The other main sections control a conversion data path. The main mechanism for control of this data path is a control shift register to which a data path pipeline is fitted, and a tap is removed in order to provide a control signal required for each stages of a data path pipeline.
[1687]An oned block has the requirements that the perfect low of data extension, i.e., the operation about the group of 8, can be started. Although invalid data (Gaps) cannot be processed in low middle, it guarantees that the operation of izz and tram is outputted as a sequence without interruption of a perfect data block of the effective extended value of 64 as a matter of fact.
[1688]B.9.9.4.1 "Conversion data path"
The fine structure of a conversion data path and t dp were illustrated in Drawing 150 before. It is cautious of not illustrating some details (for example, crocking, a shift, etc.). However, this diagram is illustrating how a data path operates it simultaneously about four value in every stage of a pipeline. an operation and a latch resource are [a fundamental sub structure of a data path, i.e., three main sections, (for example, PURIKOMON and commonness -- post -- common)] needed -- it can take [it can be alike and] and see. the pipeline latch (and addition/subtraction selector) which sets the nominated control signal in order by decoding of the control shift register State, and is carried out -- it is enabling of business. Each pipeline stage is cautious of their being actually four clock cycles in length.
[1689]Inputs are collected in a conversion data path, an intermediate result is memorized to a pipeline, and there is a latch stage of many with required arranging an

output in order. Some latches are loaded by a Mac SHINGU type that is, and they may be conditionally loaded from one or more sauce. All the latches have a jam, and the separate clock and enable input which are possible-ized type things. This means that it is easier to generate an enable signal to right timing rather than must take into consideration the problem which will be produced when the made clock scheme is applied of being irregular.
[1690]The main operational elements needed are as follows.
[1691]- Many fixed coefficient multiplication machines (carry save output)
- a carry save -- the operation of all an adder carry save subtractor, decomposition adders and decomposition adders / subtractors is performed by two's complement display. This may be the usual gestalt (decomposed) or a gestalt (two numbers which are got blocked and display value with the actual sum total) of a carry save. All the numbers are decomposed before memory, and since this is the most expensive operation from a point of time, only one decomposition operation is performed for every pipeline stage. Decomposition operation is performed here and all the operations use an easy influencing carry. Although this has a dramatically small resolver, a comparatively late thing is meant. Since decomposition governs the whole time in each stage, there is an opportunity to speed up the whole conversion using a high-speed decomposition arithmetic unit clearly.
[1692]B.9.9.5 ""Ras"-rounding and SACHU ration"
In this invention, a ras block chooses a 22-bit fixed-point number from the output of the 2nd-dimensional oned, and achieves work of changing these into the 9-bit required signed integer result of having gone around correctly and having been saturated. This block carries out the work divided by 2 [required in order to compensate the PURISUKE ring of =2 which carries out the work further divided by peculiar required 4 in a scheme (2/N term), and also is carried out in two-dimensional each]. It is interpreted as the work divided by this 8 being the triplet left behind more than the fixed-point position was predicted, that is, the implication of processing that result as a 15-bit integer display and a 7 (4-bit fraction not but rather) bit fraction is carried out. The rounding mode performed is a thing "which go around a course to a positive infinity" and for which it is got blocked and 1 is correctly added to the fraction of 0.5. Since this is in easiest rounding mode for performing, it is performed fundamentally. This result is inspected in order to see the signed result which is 9 bits whether saturated within the limits of this at the maximum or the minimum value, after rounding (conditional increment of an integral part) is completed. This is performed by the inspection of the increment carried out with the high order bit of the original integer value.
[1693]A token stream is analyzed and a round and SACHU ration operation are applied only to a data token extension value so that it may be usual. The depth has a pipeline of two stages and a block is thoroughly performed by zcells.

[1694]B.9.9.6 "Idctsels -IDCT register selection decoder"
This block is an easy decoder which decodes four microprocessor interface address lines and a sel test input on the selection line for each block test access (SUNOPA and RAM). A block comprises only zcells joint logic. The selection decoded is shown in Table 227.
[1695]
[Table 225]
[1696]
[Table 226]
B.9.9.7 "Idctregs-IDCT control register and event"
This block of an invention includes the example of standard event logic block which processes the single memory map finishing bit vscan which can be used in order to make an izz rearrangement change so that the insufficient error, superfluous error, and IDCT output of data may be scanned vertically. This bit is reset by value "0", that is, the scan-out of the default mode is carried out horizontally. In order to form the idctevent signal which can be used as interruption, both OR of the two possible events is carried out. Refer to the section B.9.10 about the address and bit position of a register and an event.
[1697]B.9.9.8 "Clock generation machine"
In IDCT, a two "standard" type (clkgen) clock generation machine is used. This is carried out as there are two separate scan passes. A clock generation machine is called idctcga and idctcgb. Functionally, the only difference is that idctcgb does not need to generate a notrstl signal. The buffering quantity for [each] the clock in two clock generation machines and a reset output is separately adjusted so that actual loading driven by each clock or reset may be suited. Suiting loading was actually measured from the gate and track capacitance of the last layout.
[1698]When Block Place and Route (BPR) of an IDCT top level is performed, an advantage, the clock (ph0 b and ph1 b) by which load was more heavily carried out since these tracks carried active current -- in order to make the width of recording track of the section of the beginning of the clock distribution tree of business increase, it was pulled out from the capability of the interactive global-area routing feature.
[1699]B.9.9.9 "JTAG control block"
Since IDCT has two separate scan chains and two clock generation machines, there are two cases in standard JTAG control block and jspctle. These connect between a test port and two scan passes.
[1700]B.9.10 "Event and control register"
IDCT can generate two events and has one control bit. When an inaccurate data token is detected, it is generated by the decheck block before IDCT, and two events are, and are dct too fewevent and idct too manyevent. One control bit is vscan set up when it is required to operate IDCT with the output scanned vertically. Therefore, this bit

controls an izz block. All the event logic and MIMORI map control bits are placed into the block idctregs.
[1701]From a viewpoint of IDCT, these registers are put on the next location. A tri-state i/o wire is n derrd, and in these locations, suitably, a lead and in order to carry out a light, n serrd is used.
[1702]
[Table 227]
[1703]
[Table 228]
B.9.11 ″Execution″
B.9.11.1 ″Logic design approach″
According to the invention, there is a unified easy logic design strategy and, probably, it as used in the design of all the IDCT blocks meant that it is possible to perform a ″safe″ design by the quick clear method. The easy scheme which uses only a master slave for much control logic was adopted. Only asynchronous set/reset input were connected to the right system reset. In order to carry out the same function more efficiently, possibly it was also possible to have provided clever non-standard type circuitry, but this scheme has the following advantages.
[1704]- notionally simple - - glitch which - operating speed's which is easy to design is quite clear (latch -> logic -> latch -> refer to the design of a logic style), and follows automatic analysis is not a problem (see the SR latch)
- - on which - scan pass which uses only a system reset for initial setting enables it to act correctly -- there are many places where the latch transparent d-type which enables the C code generating which follows automatically was used, and these were described below.
[1705]B.9.11.1.1 ″2-wire system interface latch″
A standard block structure uses a latch for the 2-wire system interface of an input and an output. Logic does not exist between an output 2-wire system latch and the input 2-wire system latch following the next.
[1706]B.9.11.1.2 ″ROM interface″
In the IZZ sequence generator in the output of ROM, a latch is used on account of the requirements for timing of a ROM circuit.
[1707]B.9.11.1.3 ″Conversion data path and control shift register″
Although it is possible to mount all pipeline storage stages as a perfect master slave device, there is serious saving obtained by using a latch on account of the storage capacity needed. However, this scheme is urging it for a user to take some elements into consideration.
[1708]- Since a control shift register is used as enabling, it must make the control signal of both phases (the necessity of getting it blocked and using a latch in this shift register).

- Since the latch of -1 ** who becomes complicated by latch's use outputs timing analysis to another latch of the same phase, t In spite of probably postc not making an edited code automatically any longer (on account of the timing of enabling, this is not a problem of a circuit), the area saved by latch's use makes it that worthy to accept these elements in this invention.
[1709]B.9.11.1.4 ″Microprocessor interface″
In the keyhole logic an event, the register block idctregs, and for RAM cores, there are requirements for a latch (and ricin clo NAIZA) for the character of this interface.
[1710]B.9.11.1.5 ″JTAG test control″
These standard blocks use a latch.
[1711]B.9.11.2 ″Problem of a circuit design″
Work in which it succeeded in the design of the library cell used in IDCT designs (a standard cell, a data path library, RAM, ROM, etc.) is set aside, and there are no requirements for a transistor level circuit design in IDCT. (Hspice is used) The circuit simulation was performed out of some publicly known critical paths in the conversion data path, and in the path near the maximum length permitted further, Hspice was used also in order to check the result of a critical path analysis (CPA) tool.
[1712]in operation usual in IDCT -- perfect -- being static (it can be got blocked and we can stop a system clock indefinitely) -- when a test clock is suspended (or a low speed is used dramatically), it declines -- I will come out -- the latch who can scan has a dynamic node -- cautions. Because of the non-reproducibility of some nodes which present Vt descent (for example, Max output), probably, IDCT is not ″micropower″, when static.
[1713]B.9.11.3 ″Layout approach″
The holistic approach to layout execution of this invention was using BPR (some manual interference), in order to arrange perfect IDCT which comprised many zcell(s) and a small number of macro block. These macro blocks were the layouts (for example, RAM, ROM, a clock generation machine, a data path) by which manual edit was carried out, or, in the oned block, were built from further zcell and a data path using BPR.
[1714]The data path was built from the kdplib cell. Besides, in addition, the layout variation to which the kdplib cell was limited locally was defined, and it was used when this was accepted to provide a worthy size benefit. The data path used in each oned block and oned d is the one greatest element in a design exceptionally.
Remarkable efforts were paid in order to optimize the size (height) of this data path.

[1715]Since exact ordering of the element in a data path affects the method by which a link is processed, the organization of a conversion data path and t dp are decisive rather. Since there is a maximum-permissible value (10 is [eight] ideally possible although it is very inconvenient), it is important to make into the minimum the number of overs(es) (vertical wire which is not connected to a subblock) generated on the

point which crowded most. The split of the data path is logically carried out to three main subsections, and a data path layout is performed in this way. In each subsection, there are many methods of there being four data flow arranged in parallel actually (it should combine on various points), therefore organizing data flow (and position of all the elements) within each subsection. In order to make it possible to attain the layout which will be connected correctly, assignment in ordering of the block in each subsection and the physical bus pitch of a logic bus was carefully computed, before the layout started.
[1716]B.9.12 "Collation"
In many levels, it succeeded in collation of IDCT from collation of the top level of an algorithm to the last layout check.
[1717]The initial work about conversion structure was done in C, and perfect accuracy and a bit exact integer model were developed. In order to prove the conformity to an H.261 accuracy standard and to measure the dynamic range of calculation within conversion structure, various tests were carried out about the bit exact model.
[1718]By writing M in many cases, the design advanced behavioral science explanation (for example, control of a data path and RAM) of the subblock. This **** explanation was simulated in Lsim, before moving to the design of rough explanation of the block. In a certain case (for example, RAM, a clock generation machine), behavioral science explanation was used also for the top-level simulation.
[1719]The strategy for carrying out a logic simulation was simulating a schematic diagram for all the things which will be appropriately simulated on the level. Since this result was a far small quick simulation, the library cell (getting it blocked zcell and kdplib) of a low level was simulated mainly using that behavioral science explanation. Besides, in addition, a behavioral science library cell provides the feature of the timing check which can emphasize the structural problem of a certain circuit. A certain simulation was carried out using transistor description of a library cell as a reliance check. It had intention of all the logic simulations being performed by the zero delay method, therefore comparing the performance on a function. Collation of action of real timing was performed using other art.
[1720]Although the Lsim (using RC Timing mode) switch level simulation was performed as partial collation of timing performance, the check for the problem (for example, glitch high sensitivity circuit) of other potential transistor levels is provided.
[1721]The main collation art for checking a timing problem was use of the CPA tool and the path option for datechk. It was used in order that this might identify a longer signal path (a certain thing was already known), and Hspice was used in order to compare CPA analysis, when [a certain] serious.
[1722]Since the bulk of the IDCT act was trained by the flow of the token through the device, almost all the Lsim(s) simulation was performed by standard sauce -> block ->

sink methodology. An additional simulation is also required in order to test the feature [AKUSASU / feature / through a microprocessor interface (composition, an event, and test logic)], and the test feature accessed via JTAG/scan.
[1723]An edited code simulation may be easily attained by the person skilled in the art using many of same token streams too used in the standard sauce -> block -> sink method and Lsim collation for all the IDCT(s).
[1724]B.9.13 "Testing and test support"
This section treats the mechanism with which analysis of how the object for testing and each block are tested is provided.
[1725]To test access. : whose three mechanisms provided are as follows. - Micro-processor access and control to the micro-processor access snow pub rock to a RAM core, and the scan pass access IDCT to data path logic have two "SUNOPA" blocks and one "sault pass NOPA" block. Drawing 149 shows test access of the position of snow pub rock, and other microprocessors.
[1726]It is the purpose of testing those actions about a token flow using these and two RAM blocks, and it is possible to isolate each main blocks. It is possible to control the token input to any blocks and to observe the token port output of the block isolated using micro-processor access. There are two separate scan passes which pass the flip-flop of all in the control section (almost) of each block and a latch, and pass some data path latches in the case of an oned conversion data path pipeline. Two scan passes are displayed as a and b, the former moves to an ipfmt block from a decheck block, and the latter moves to a ras block from the first oned block.
[1727]Access to SUNOPA is possible by accessing the suitable location by which the memory map was carried out by the usual method. (Using a ramtest input as a suitable thing) It can say that it is the same also in the case of a RAM core. A scan pass is accessed through a JTAG port by the usual method.
[1728]Each block is discussed in relation to various test problems.
[1729]B.9.13.1 "Decheck"
This block has the standard structure (see Drawing 148) in which two latches for the 2-wire system interface of an input and an output surround a processing block. Since these 2-wire system latches only pass data always and do not have the depth of the logic tested when becoming possible so that it may be usual, any scans are not performed to a 2-wire system latch. In this block, "control" section comprises 1 stage pipeline of zcell which has all on the scan pass a. the logic in a control section is comparatively simple, and most complicated paths come out in generating of the data extension count for which a 6-bit increment machine is probably used.
[1730]B.9.13.2 "Izz"
This block is modification of standard structure and contains the RAM core block added to a control section with a 2-wire system interface latch. A control section is mounted by zcell used for address sequence generating, and small ROM. All the

zcell(s) are on the scan pass a, and there is data through access and a zcell latch in a ROM address. There is logic for logic, for example, the generating plus increment of a number, or the capability to decrease. Besides, in addition, there is a 7-bit full adder used for read address generating. A RAM core lets a keyhole register pass and it is accessible via a microprocessor interface in it. Refer to Table 225 and Table 226.
[1731]B.9.13.3 ″lp fmt″
This block also has standard structure. Although control logic is mounted by quite easy zcell logic (all are on the scan pass a), since logic here is very shallowly simple, in a data path, latching of data, and shifting / Mac SHINGU are performed without direct access.
[1732]B.9.13.4 ″Oned″
Here, this block is divided into random logic and a data path section according to standard structure. zcell logic is comparatively clear and all the zcell(s) are on the scan pass a. The control signal for conversion pipeline data paths is pulled out from the long shift register which comprises a zcell latch on a scan pass. Besides, in addition, some pipeline latches are on a scan pass, and this is performed from the logic between some of pipelines' stages (for example, a multiplier and an adder) being quite deep. A non-data token is sent along with a shift register, it is mounted as a data path, and test access to any of these stages does not exist, either.
[1733]B.9.13.5 ″Tram'″
There is a close resemblance between this block and an izz block. However, there is no ROM used in address sequence address generation in this case. This is performed in algorithm. All the zcell control State is on the data path b.
[1734]B.9.13.6 ″Rras'″
This block is thoroughly performed by zcell according to standard structure. The most complicated logic function is an 8-bit increment machine used at the time of a rounding rise. Other logic of all the is quite simple. All the State is on the scan pass b.
[1735]B.9.13.7 ″-- others -- top-level block″
There are some other blocks which appear in the top level of IDCT. SUNOPA is also a part of test access logic clearly like being JTAG control block. There are two clock generation machines without special test access (however, they are supporting the various test features). The block idctsels is the combination zcell logic for carrying out decoding of the micro-processor address, and the block idctregs contains a micro-processor accessible event and an IDCT related control bit.
[1736]Section B.10 ″Preface″
B.10.1 ″View of a time decoder″
The internal structure of the time decoder by this invention is shown in Drawing 151.
[1737]All the data flow (and much data flow within a block) between chip blocks is controlled by a 2-wire system interface (for details, see a technical reference book and the section), and each arrow in Drawing 151 expresses a 2-wire system interface.

The token stream which enters passes the input interface which synchronizes the data from an external system clock with the internal clock pulled out from a phase-locked loop (ph0/ph1). He follows to an address generator the;1 ** stream by which the split of the token stream is carried out next to two paths via a top fork, and follows other streams to 256-word FIFO. While FIFO carries out the buffer of the data, front I or the data from p frames is pulled out from DRAM, and before being added to the error-of-input-data data from the space decoder in a prediction adder (P and the B frame), the process of it is carried out in a prediction filter. As an output frame is in a right order between MPEG decodings, frame rearrangement data must be pulled out for I and p frames. The relocated data is inserted in the stream within a lead rudder block.

[1738]An address generator generates an address separate for a forward and backward prediction, rearrangement, a lead, and write back, and the split of the data by which write back is carried out is carried out from the stream within a light rudder block. Finally, resynchronization of the data is carried out to the external clock within an output interface block.

[1739]All the main blocks in a time decoder are connected to an internal microprocessor interface (UPI) bus. This is pulled out from the external microprocessor interface (MPI) within a microprocessor interface block. This block has address decode for various blocks in the chip relevant to it. Event logic unites with a microprocessor interface.

[1740]The remaining logic of a time decoder is test-related fundamentally. To the JTAG boundary scan feature, 1149.IEE1 (JTAG) interface provides the 1st with the interface to an internal scan pass, while. The 2-wire system interface stage which enables intrusive access to data flow via a microprocessor interface between test modes is contained [2nd] in the strategy point within pipeline structure.

[1741]Section B.11 "Crocking, a test, and related problem"

B.11.1 "Clock form"

Probably, it will be useful to recognize the clock forms and those relations in a chip, before taking into consideration each functional blocks in a chip.

[1742]Between the usual operations, almost all blocks of a chip move synchronizing with the signal pllsysclk from a phase-locked loop (PLL). The exception over this is a DRAM interface, that timing is governed by the necessity of synchronizing with an iftime subblock, and this subblock generates a DRAM control signal (notwe, notoe, notcas, notras). The clock of the core of this block is carried out with the 2 phase non overlapping clocks clk0 and clk1, and they are pulled out from Quadra CHUA 2 phase clock separately supplied from PLL cki0, cki1 and clkg0, and ckg1.

[1743]Since clk0 and a clk1 DRAM interface clock synchronized with the clock in the remaining chip, in the DRAM interface and the interface during the remaining chip, the measures for removing the possibility of a metastable (in practice as much as

possible) act were taken. In the output interface of :address generator by which synchronization is two fields and it is generated () [addrgen/predread/psgsync and] addrgen/ip wrt2/sync18 and addrgen/iprd2/sync18, and in the block which controls ″swinging″ of swing buffer RAM in a DRAM interface, come out and it is (refer to the section about a DRAM interface). In each case, a synchronization process is attained by three serial metastable hard flip-flops. What should be careful of is that this means that clk0/clk1 is used in the output stage of an address generator.
[1744]In addition to these clock forms that synchronized thoroughly, there is a clock generation machine with separate many made to generate a 2 phase non overlapping clock (ph0, ph1) from pllsysclk. The address generator, the prediction filter, and the DRAM interface have an own clock generation machine respectively, and the chip of; remainder separates and is moved from a common clock generation machine. Two portions are one of such reasons. The capacitive load applied to each clock generation machine the 1st is decreased, and a smaller clock driver and clock routing width which decreased are made possible. Each scan pass is controlled by a clock generation machine by the 2nd, therefore can use a short scan pass now for it by the increase in the number of clock generation machines.
[1745]It is required to carry out resynchronization of the signal which crosses these clock form boundaries and is moved. It is because the skew which is not [between the non overlapping clocks pulled out from a different clock generation machine] important probably means that the negative lap occurred in the interface. Each ″SUNOPA″ The circuit built in the block (see section B.11.4) guarantees that such a thing does not occur, and it snow pub rock, Except for the address generator front with which resynchronization is performed in token decode blocks, it was put on the boundary between all the clock forms.
[1746]B.11.2 ″Control of a clock″
Each standard clock generator generates the clock with which many it is made to be possible [whose operation] differ by the usual mode and scanning test mode. Although other sections explain control of the clock in a scanning test mode in detail, some of clocks by which what should be observed is emitted with a clock generation machine (tph0, tph1, tckm, tcks) are not usually added to the primitive sign of a schematic diagram. This is because a scan pass is automatically made by the post processor which connects these clocks correctly. If it sees from a functional viewpoint,; act which can disregard the fact that the post processor connected a different clock from what was shown in the schematic diagram is the same.
[1747]During the usual operation, a master clock can be pulled out by the way many differ. Table 229 shows signs that various modes can be chosen by the State of the pin pllselect and an override.
[1748]
[Table 229]

B.11.3 ″2-wire system interface″
In a technical reference book, the overall functionality of a 2-wire system interface is explained in detail. However, a 2-wire system interface is used for communication during all the blocks in a time decoder, almost all blocks comprise many pipeline stages, and the all are 2-wire system interface stages. Therefore, in order to be able to interpret many schematic diagrams, it is important to understand internal mounting of a 2-wire system interface. Generally, these internal pipeline stages are constituted as shown in Drawing 152.
[1749]Drawing 152 shows the latch logic latch display, and this is the arrangement by which normal use is carried out. However, when many stages are placed together, it is also effective to consider a ″stage″ to be latch latch logic (mode in which it is known well for many engineers). Also when all the interblock communications send a block and they receive use of latch logic latch arrangement, it can be a latch versus a latch, without inserting logic in between.
[1750]The same with removing logic block, connecting a valid signal with data among latches directly in Drawing 152, again, and the gate of out valid and out accept being carried out, By connecting to an inaccept latch in valid directly latched to the NOR gate on an input, an easy 2-wire system interface FIFO stage can be built. Data and a valid signal are spread when the accept signal corresponding to the next is high. Even if out accept reg is low by acting as OR Inge of in valid by out acceptreg by the illustrated method, data is accepted when in valid is low. By this method, when generated by the stole (a low signal is accepted), gaps (data with an effective low bit) is removed from a pipeline always.
[1751]As shown in Drawing 152, logic block is inserted and in accept and out valid can be subordinate to the State of data or a block. It is a standard that all the State within a block is held in the illustrated arrangement in the master slave device provided with the master made possible by ph1 and the slave made possible by ph0.
[1752]B.11.4 ″Snow pub rock″
In various points in a chip, snow pub rock enables access to a data stream via a microprocessor interface. There are two types of snow pub rocks. Only in the test mode by which a clock may be controlled directly, it is accessible in ordinary SUNOPA. Also while the clock is moving, it is accessible, and ″sault pass NOPA″ includes the circuit which makes an internal chip clock synchronize the asynchronous data from a micro processor bus. Table 230 has indicated the location and type of all the SUNOPA in a time decoder.
[1753]
[Table 230]
The details about use of both SUNOPA are indicated into a test section. The details of the operation of a JTAG interface are written in a JTAG document.
[1754]Section B.12 ″Functional block″

B.12.1 "Top fork"
According to this invention, a top fork achieves two different functions. The :1 ** which branches [1st] a data stream to two different streams provides the starting means and stopping means of a chip so that other one can be arranged to an address generator and it can arrange [2nd] a chip to FIFO.
[1755]The aspect of affairs of the fork portion of a component is dramatically easy. The same data is displayed on an address generator and FIFO, and before acceptance is returned to a front stage, it must be accepted by both blocks. Thus, valids of two brunches of a fork is subordinate to the acceptance from other brunches. When a chip is in a halt condition, valids to both brunches is kept low. in accept upgrades a chip with the state where it is kept low until an arrangement bit is set up highly. This guarantees that any data is not accepted until a user will arrange a chip. When users are others and a chip needs to be arranged, a user sets up an arrangement bit, and he has to wait until a chip will complete the present stream. If :1 arrangement bit whose stopping process is as follows is set up, any data will not be accepted after a flash plate token is detected by a top fork.
[1756]2) Probably, the chip has completed processing of a stream, when a flash plate token reaches a lead rudder. This makes the signal seq done high.
[1757]3) If seq done becomes high, the event bit which a microprocessor can read will be set up. The mask of the event signal may be carried out by event block.
[1758]B.12.2 "Address generator"
In this invention, an address generator (addrgen) counts the block count in a frame, and it is responsible for generating the right address sequence for DRAM-data transmission. The input of an address generator is a token stream from a token input port (a top fork is passed), and the output to a DRAM interface comprises the address and other information which are controlled by the request / acknowledgement protocol.
[1759]The request for conversion / prediction transmission to the address offset of generating and motion vector data of :, token decoding block counting, and a DRAM block address which is as follows [section / of an address generator / fundamental]. And an address generator, a rearrangement read address generator, and write address generator B.12.2.1 "Token decoding (tokdec)"
In a token decoder, the token, the frame, the block information, and the motion vector which unite to a coding standard are decoded. The information extracted from a stream is memorized by a series of registers, and it may be accessed via upi. Detection of a data token header is signed by the following block, and makes block counting and address generation possible. While moving JPEG, nothing happens.
[1760]The token list coding standard data and DEFINE MAX SAMPLING-DEFINE SAMPLING-HORIZONTAL MBS-MVD BACKWARDS-MVD decoded. FORWARDS, a picture start picture type, and prediction Mohd -- further, combining the information

from a request generator, this block controls toggling of a frame pointer and carries out the stole (damping) of the input stream. If a new frame appears in an input (with gestalt of a picture start token), the stole of the stream will be carried out, but the write back or the rearrangement lead relevant to a front frame is imperfect.
[1761]B.12.2.2 "Macro block counter (mblkcntr)"
The macro block counter of this invention comprises four fundamental counters which point out the horizontal position and vertical position of a macro block in a frame. In the beginning of time, all the counters are set as 0 on the occasion of each picture start. If a data token header arrives, the increment of the counter will be carried out and it will be reset according to the color component number within a token header and the frame structure. This frame structure is explained by the sampling register in a token decoder.
[1762]For a predetermined color component, counting advances as follows. The increment of it is carried out to each of a data token with the same new ingredient, and a level block count is reset after that until it reaches the width of a macro block. The increment of it is carried out by this reset, and a vertical block count is reset after that by it until it reaches the height of a macro block. This generating will wait for the following color component. therefore, this sequence -- macro block - it is repeated to each of the ingredient in the size vertical to coming out and a macro block being level which probably changes with each ingredients. The count will progress without the error to the following ingredient, even when little block is received to a certain ingredient rather than expected.
[1763]When less than the value from which the color component of the data token was expected, the increment of the level macro block count is carried out. (Such a thing will generate the counter, since it expects the higher component index, also when the block count of the more than expected to the predetermined color component appears.) This level count is reset when a count reaches the picture width in a macro block. This reset carries out the increment of the vertical macro block count.
[1764]It has the ability to count the macro block in an H.261 CIF format. In this case, there is a hierarchy of the special level between the pictures called a macro block and a block group. This is 11 macro-block width and the depth of three macro blocks, and a picture is always 2 group width. A token decoder pulls out a CIF bit from a picture type token, sends this to a macro block counter, and directs to count a block group. For every ingredient, the above reactions will be triggered, when there is little block count or there is. [too much]
[1765]B.12.2.3 "Block calculation (blkcalc)"
Block calculation changes the coordinates of the block in a macro block and a macro block into the coordinates for positions of the block in a picture, that is, knocks out a layer level. Of course, this must take into consideration the sampling rate of a

different color component.

B.12.2.4 "Base-block address (bsblkadr)"

The information from blkcalc is used in order to calculate the block address in a linearity DRAM address space with color component offset. Intrinsically, a linearity block address is the number of vertical block count x picture width + horizontal blocks because of a predetermined color component. This is added to color component offset, in order to form a base-block address.

[1766]B.12.2.5 "Vector offset (vec pipe)"

The motion vector information displayed by the token decoder is a gestalt of horizontal and vertical pixel offset coordinates. that is, the block formed in the block with which it is predicted for [each] the forward and the backward vector to half a pixel transposition -- giving (x, y) -- it is. It is cautious of these coordinates being positive or negative. They are first estimated according to the sampling of each color component, and they are used in order to form a block and new pixel offset coordinates. The block with which a shade part is formed is expressed in Drawing 154. The outline of a dotted line is a block with which it is predicted that it comes from there. A big arrow shows the horizontal and vertical vector to the DRAM block containing the origin of a block offset prediction block, and this case (1, 4). A small arrow shows the position of the prediction block origin in the DRAM block of new pixel offset **. Since the DRAM block is 8x8 bytes, pixel offset seems to be (7, 2).

[1767]Multiplier array vmarrla changes block vector offset into linear vector offset next. Pixel information is sent to a prediction request generator as coordinates (x, y) (pix info).

[1768]B.12.2.6 "Prediction request"

A frame pointer, a base-block address, and vector offset are added, and form the address pulled out from DRAM (lnblkad3). Only one request will be emitted if pixel offset is 0. if there is offset in either x or y dimension -- two request original block addresses and immediately the right -- or either of the things of right under - is emitted. Four requests will be emitted if both x and y have offset. The synchronization between a chip clock form and a DRAM interface clock form breaks out between the 1st addition (lnblkad3) and the State machine which emits a suitable request. Thus, the clock of the State machine (psgstate) is carried out with a DRAM interface clock, and the scanned element forms a part of DRAM interface scan chain.

[1769]B.12.2.7 "Rearrangement read request and write request"

Here, since pixel offset is not contained, each address is formed by adding a base-block address to a related frame pointer. Since prediction and data are returned to other frame storages, a rearrangement lead uses the same frame storage. Since each block has the tendency to delay prediction transmission in the address which corresponds a lead and transmission of right data, short FIFO is provided in order to memorize an address. (This is because it interacts with the stream to which a

lead/right data met chip data flow from prediction data.) Each block includes synchronization between a chip clock and a DRAM interface clock further.
[1770]B.12.2.8 "Offset"
DRAM is arranged as two frame storages and each contains three color components. A frame storage pointer and the color component offset in each frame must be programmed via upi.
[1771]B.12.2.9 "SUNOPA"
Between : and blkcalc by which SUNOPA is arranged as follows in this invention, and bsblkadr(s) - This interface comprises the picture width within vertical block coordinates, suitable color component offset (for [that] ingredients), and a block with it being level.
[1772]- The back-base-block address of bsblkadr.
[1773]- It is the pixel offset within a block in the information about back-linearity block offset, prediction Mohd, the color component, and H.261 operation of vec pipe.
[1774]- As the paragraph of back - "prediction request" of Inblkad3 explained, since physical block-address sault pass NOPA uses it during the test of external DRAM, it is placed into a rearrangement lead and a write request generator. For details, refer to the DRAM interface section.
[1775]B.12.2.10 "Scan"
The addrgen block has a scan chain of itself and the crocking is controlled by the clock generation machine (adclkgen) of the block [itself]. It is cautious of the request generator in the back end of a block being within the limits of a DRAM interface clock form.
[1776]B.12.3 "Prediction filter"
An overall structure of the prediction filter by this invention is shown in Drawing 155. A forward and a backward filter are the same, and filter an MPEG forward / backward prediction block. Only a forward filter is used in H.261 Mohd (since an H.261 stream does not include backward prediction, there should be an h261 on input of a backward filter low eternally). An overall prediction filter block comprises a pipeline of a 2-wire system interface stage.
[1777]B.12.3.1 "Prediction filter"
Each prediction filter acts independently thoroughly [other prediction filters], and shortly after valid data appears in the input, it will process data. A prediction filter comprises four separate blocks and two are [of them] the same so that clearly from Drawing 156. Probably, it will be better to explain the operation of these blocks independently for the operation of MPEG and H.261. Since H.261 is the most complicated, it explains first.
[1778]B.12.3.1.1 "H.261 operation"
:Fi = (xi+1+2 xi+xi-1) / 4 which is as follows [type / which is used / one-dimensional filter] (i<=i<=6)

Fi = xi (other i)
This is applied to each 8x8-block low with x prediction filter, and is applied to each column with y prediction filter. Although the mechanism by which this is attained is illustrated to Drawing 157, it expresses a pfltldd schematic diagram fundamentally. A filter comprises three 2-wire system interface pipeline stages. The registers A and C are reset for the pixel of the low beginning and the last, and data is passed with no changing the register B, D, and F (the contents of B and D are added to 0). Control of Bx2mux is set so that the output of the register b may be shifted to the left only 1. This shifting is added to one place always shifted in every event. In this way, 4 is hung on all the values (it is more than this behind). For other pixels of all the, xi+1 is loaded to the register C and the register B and xi-1 are loaded for xi to the register A. As shown in Drawing 157, an H.261 filter type is performed after that. Since vertical filtering is carried out in three level groups (see the note about the following dimension buffer), it is not necessary to process the pixel of the beginning in a low, and the last separately. Control and counting of the pixel in a low are carried out by the control logic relevant to a 1-dimensional each filter. It should be cautious of the result not being divided by 4. After horizontal and vertical filtering is performed so that the accuracy on an operation may not be lost, in the input of a prediction filter adder (section B.12.4.2), the work divided by 16 (only 4 is shifted to the right) is done. The registers DA, DD, and DF send control information to a pipeline. This contains h261 on and last byte.
[1779]The function of formatting is ensuring data only being expressed in x-filter as a right order among other blocks found in a prediction filter. The shift register of three stages is only required for this, the 1st stage is connected to the input of the register C, the 2nd stage is connected to the register B and the 3rd stage is connected to the register A so that the above may show.
[1780]Between x filter and y filter, a dimension buffer buffers data and the group of three vertical pixels is displayed on y-filter. Although these three groups are still processed horizontally, any transposition is not generated within a prediction filter. In relation to Drawing 158, a pixel shows the sequence outputted from a dimension buffer in Table 231 and Table 232.
[1781]
[Table 231]
[1782]
[Table 232]
B.12.3.1.2 "MPEG operation"
:Fi = (xi+xi+1) / 2 which performs half a pixel interpolation with an easy prediction filter between MPEG operations (0<=i<=8, half a pixel)
Fi = xi (0<=i<=7, integer pixel)
If an h261 on input is not low, this is default filter operation. If the signal dim to a

one-dimensional filter is low, integer pixel interpolation will be performed. Therefore, if h261 on is low and xdim and ydim are low, all the pixels will be sent directly, without filtering. When the dim signal to a one-dimensional filter is high, it is a clear necessary condition that a low (or column) becomes 8 pixel width (or more than it). This is summarized in Table 233.
[1783]
[Table 233]
In Drawing 157 and a ″one-dimensional prediction filter″, the operation of a one-dimensional filter is for MPEG INTAPIKUSERU like being a sake of the pixel of the beginning in the low of H.261, and the last. For half a pixel of MPEG operation, the register A is reset eternally, and the output of the register C is shifted to the left only 1 (the output of the register B is always shifted to the left only 1). In this way, after two clocks, although it is 4 times the result with the register F required for it including (2B+2C), as for this, the number which passed x filter and y filter is processed in the input of the prediction filter adder from which only 4 is shifted to the right.
[1784]The function of formatting and a dimension buffer is simple also in MPEG. ; dimension buffer which the formatting must collect two effective pixels and must be sent to x-filter for half a pixel interpolation of them only needs to carry out the buffer of the one low. Since filtering operation changes 9-pixel low into 8-pixel low after data passes x-filter, deserving attention is that only eight pixels exist in a low. The pixel ″lost″ is replaced about the gap in a data stream. When carrying out half a pixel interpolation, x-filter inserts a gap in the end of each low (after eight pixels each), and a;y-filter inserts the gap of eight pieces in the end of a block. It is this in the end of a block, and it is important from aligning with other tokens between the data tokens in the stream by which eight pieces or nine gap groups come from a data token header and FIFO. This suppresses the case of being the worst, through the chip generated when the block of 9x9 is filtered to the minimum.
[1785]B.12.3.2 ″Prediction filter adder″
Between MPEG operations, an early picture, a picture of the second half, or both average is used, and prediction is formed. The prediction formed from an early frame is called forward prediction, and the prediction formed from a frame of the second half is called backward prediction. The function of a prediction filter adder (pfadd), It is determining any shall be passed between which filter finishing predicted value being used, a forward [or] (a forward, backward or its both), backward filtered prediction, or both average (it goes around a course toward a positive infinity).
[1786]prediction Mohd is between blocks -- that is, the time of an upgrade -- or after fwd 1st byte which directs the byte of the last of the present prediction block and/, or a bwd 1stbyte signal becomes activity, it can only change. If the present block is forward prediction, only fwd 1st byte will be investigated. If it is backward prediction, only bwd 1st byte will be investigated. If it is prediction of two-way-type tropism, fwd

1st byte and bwd 1st byte will be investigated.
[1787]It is determined which predicted value the signals fwd on and bwd on use. At any time, even if both these signals are activity, it is good for both not to be activity. If there is a gap when data effective in the time of start-up or the input of a block does not exist, a block will go into the State, when neither of the signals is activity.
[1788]The signals fwd ima twin and bwd ima twin which direct any of :forward block or the backward block with which two standards are used for determining prediction Mohd for the following block are a part of prediction pairs of two-way-type tropism. And they are the buses fwd p num [1:0] and bwd p num [1:0]. These buses include the number which carries out the increment only of 1 for each new prediction block or a prediction block pair. When there are two forward prediction blocks and a prediction block of the two-way-type tropism following it for example, these being blocked so that it may reach before the 2nd forward prediction block at the input of a prediction filter adder, It is because the DRAM interface can pull out the backward block of bidirectional prediction enough before. Similarly, other sequences can be backward and forward prediction can also come out of a sequence in the input of a prediction filter adder. Thus, the :1 effective forward data determined as follows exists, and if next prediction Mohd has high fwd ima twin, The block which carries out the stole of the block until effective backward data arrives with a bwd ima twin set, and equalizes each prediction value pair is passed.
[1789]2) Effective backward data exists, if bwd ima twin is high, the stole of the block will be carried out until effective forward data arrives with a fwd ima twin set, and progress as mentioned above. A stole will not be performed if both a forward and backward data are effective.
[1790]3) Although effective forward data exists, if fwd ima twin is not set up, fwd p num is investigated. If this is equal to the front (pred num memorizes) number from prediction+1, prediction Mohd will be set as a forward.
[1791]3) Although effective backward data exists, if bwd ima twin is not set up, bwd p num is investigated. If this is equal to the front (pred num memorizes) number from prediction+1, prediction Mohd will be set up backward.
[1792]Before the early valid signal from 1 stage back in a pipeline is used and the data of the beginning from a new block arrives, it is cautious of the ability of prediction filter adder mode to be set up. This guarantees that any stoles are not introduced into a pipeline.
[1793]With the filtered data, ima twin and a pred num signal are not sent along with a forward and a backward prediction filter pipeline. : which it is from the reason of the following [this] -- one -- the signal of these is investigated only when fwd 1st byte and/, or bwd 1st byte is effective. Thereby, the triplet pipeline stage of about 25 can be saved in each prediction filter.
[1794]2) It is effective when fwd 1st byte and/, or bwd 1st byte reaches [be / it /

under / block / leading] a prediction filter adder, since a signal is still effective.
[1795]3) A signal is investigated in front of 1 clock with which data arrives anyway.
[1796]B.12.4 "Prediction adder and FIFO"
By adding the data from a prediction filter to error data, a prediction adder (padder) forms a predicted frame. In order to compensate the delay from the input which passes along an address generator, a DRAM interface, and a prediction filter, error data passes 256-word FIFO (sfifo), before reaching a padder.
[1797]A coding standard token, a prediction mode token, and a data token are decoded in order to determine when a prediction block is formed. 8-bit prediction data is added to the 9-bit two's complement error data in a data token. The result is restricted to the range of 0-255, and progresses to the following block. It is cautious of these data restrictions being applied also to all the Intra coded data containing JPEG.
[1798]The prediction adder of this invention includes the mechanism for detecting further the mismatching in the data which arrives from FIFO and a prediction filter. The data volume from a filter must be [theory top] correctly equivalent to the number of data tokens from FIFO including prediction. In the case of a serious malfunction, a padder tries recovery.
[1799]The end of the data block from FIFO and a filter is respectively marked by in extn and the f1 last input. When the filter end of data is detected before the end of a data token, the remaining tokens continue an output, without being changed. On the other hand, when a filter block is longer than a data token, all the superfluous filter data is accepted, and the stole of the input is carried out until it is thrown away.

Timeout: The process for displaying translation results will be terminated.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]
[Drawing 1]The figure showing six cycles of the six-step pipeline for the combination from which two internal control signals differ.
[Drawing 2]The figure showing the pipeline in whom each stage contains a secondary data storage in order to show the method where a pipeline stage answers the delay in a pipeline, and which can perform compression and extension.
[Drawing 3]The figure showing the pipeline in whom each stage contains a secondary

data storage in order to show the method where a pipeline stage answers the delay in a pipeline, and which can perform compression and extension.
[Drawing 4]The figure showing control of the data transfer between the stages of the desirable example of the pipeline who uses 2 wiring interface and a polyphase clock.
[Drawing 5]The figure showing control of the data transfer between the stages of the desirable example of the pipeline who uses 2 wiring interface and a polyphase clock.
[Drawing 6]The figure showing control of the data transfer between the stages of the desirable example of the pipeline who uses 2 wiring interface and a polyphase clock.
[Drawing 7]The figure showing control of the data transfer between the stages of the desirable example of the pipeline who uses 2 wiring interface and a polyphase clock.
[Drawing 8]The block diagram showing two continuous pipeline processing stages which have the fundamental example and 2 wiring transfer control of a pipeline stage into which 2 wiring interface was built.
[Drawing 9]An example of the timing diagram showing the relation between internal control signals with the timing signal used in the pipeline stage shown in drawing 8, and an I/O data.
[Drawing 10]An example of the timing diagram showing the relation between internal control signals with the timing signal used in the pipeline stage shown in drawing 8, and an I/O data.
[Drawing 11]The block diagram showing an example of a pipeline stage which maintains the state under control of an extension bit.
[Drawing 12]The block diagram of the pipeline stage which decrypts stage starting data word.
[Drawing 13]The block diagram showing use of 2 wiring transfer control in an example of a data doubleness pipeline stage.
[Drawing 14]The block diagram showing use of 2 wiring transfer control in an example of a data doubleness pipeline stage.
[Drawing 15]An example of the timing diagram showing 2 phase clock used in the example illustrated to drawing 13 and drawing 14, 2 wiring transfer control signal, and other in-house datas and control signals.
[Drawing 16]An example of the timing diagram showing 2 phase clock used in the example illustrated to drawing 13 and drawing 14, 2 wiring transfer control signal, and other in-house datas and control signals.
[Drawing 17]The block diagram of the processing stage which can be reconstructed.
[Drawing 18]The block diagram of a space decoder.
[Drawing 19]The block diagram of a time decoder.
[Drawing 20]The block diagram of a video formatter.
[Drawing 21]The memory map in which the 1st composition of a macro block is shown.
[Drawing 22]The memory map in which the 2nd composition of a macro block is shown.
[Drawing 23]The memory map in which another composition of a macro block is shown.

[Drawing 24]The Venn diagram of the table selection value which may exist.
[Drawing 25]The figure showing the variable length of the image data used by this invention.
[Drawing 26]The block diagram of a temporary decoder containing a prediction filter.
[Drawing 27]The figure showing prediction filter processing with a picture.
[Drawing 28]The common figure showing the structure of a macro block.
[Drawing 29]The common block diagram of a start code detector.
[Drawing 30]The figure which illustrates the start code in a data stream.
[Drawing 31]The block diagram showing the relation of an output latch with a flag generator, a decryption index, a header generator, and an additional word generator.
[Drawing 32]The block diagram of a space decoder DRAM interface.
[Drawing 33]The block diagram of a write-in swing buffer.
[Drawing 34]The illustration figure showing the prediction data offset from the block under processing.
[Drawing 35]The figure showing prediction data offset of (1, 1).
[Drawing 36]The block diagram showing the dividing state device of the Huffman decoder and a space decoder.
[Drawing 37]The block diagram of a prediction filter.
[Drawing 38]The figure showing a typical decoder system.
[Drawing 39]The figure showing a JPEG still picture decoder.
[Drawing 40]The figure showing a JPEG video decoder.
[Drawing 41]The figure showing a multiplex standard video decoder.
[Drawing 42]The figure showing a start and end of a token.
[Drawing 43]The figure showing a token address and a data field.
[Drawing 44]The figure showing the token on an interface larger than 8 bits.
[Drawing 45]The figure showing a macro block.
[Drawing 46]The figure showing 2 wiring interface protocol.
[Drawing 47]The figure showing the position of an external 2 wiring interface.
[Drawing 48]The figure showing clock propagation.
[Drawing 49]The figure showing the timing of 2 wiring interface.
[Drawing 50]Access structure is illustrated.
[Drawing 51]The figure showing a read-out transfer cycle.
[Drawing 52]The figure showing access start timing.
[Drawing 53]The figure showing the example of access of two write-in transmission.
[Drawing 54]The figure showing a read-out transfer cycle.
[Drawing 55]The figure showing a write-in transfer cycle.
[Drawing 56]The figure showing a RIFURESHU cycle.
[Drawing 57]The figure showing the example of DRAM (nine bit addresses) with the depth of 32 bit data buses and a 256k bit.
[Drawing 58]The figure showing the timing parameter for strobe signals.

[Drawing 59]The figure showing the timing parameter between which two strobe signals.
[Drawing 60]The figure showing the timing parameter between a bus and a strobe signal.
[Drawing 61]The figure showing the timing parameter between a bus and a strobe signal.
[Drawing 62]The figure showing MPI read-out timing.
[Drawing 63]The figure showing MPI write timing.
[Drawing 64]The figure showing arrangement of the big integer of a memory map.
[Drawing 65]The figure showing a typical decoder clock gestalt.
[Drawing 66]The figure showing the requirements for an input clock.
[Drawing 67]The figure showing a space decoder.
[Drawing 68]The figure showing input and output of an input circuit.
[Drawing 69]The figure showing a coding port protocol.
[Drawing 70]The figure showing a start code detector.
[Drawing 71]The figure showing the detected start code and the changed token.
[Drawing 72]The figure showing the place where a start code detector lets a token pass.
[Drawing 73]The figure showing the lap of an MPEG start code (byte alignment).
[Drawing 74]The figure showing the lap of an MPEG start code (non-byte alignment).
[Drawing 75]The figure showing the jump between two video series.
[Drawing 76]The figure showing insertion of an additional token.
[Drawing 77]The figure showing decoder start control.
[Drawing 78]The figure showing the enabled stream by which cue was carried out before the output.
[Drawing 79]The figure showing space decoder buffers.
[Drawing 80]The figure showing a buffer pointer.
[Drawing 81]The figure showing video demulti PUREKKUSA.
[Drawing 82]The figure showing picture structure.
[Drawing 83]The figure showing the structure of 4:2:2 macro blocks.
[Drawing 84]The figure showing how to calculate from the size of the size pixel of a macro block.
[Drawing 85]The figure showing a space decoder.
[Drawing 86]The figure showing a general view of H.261 inverse quantization.
[Drawing 87]The figure showing a general view of JPEG inverse quantization.
[Drawing 88]The figure showing a general view of MPEG inverse quantization.
[Drawing 89]The figure showing a quantization table memory map.
[Drawing 90]The figure showing a general view of JPEG baseline system structure.
[Drawing 91]The figure showing the JPEG image by which the token was carried out.
[Drawing 92]The figure showing a time decoder.

[Drawing 93]The figure showing picture buffer specification.
[Drawing 94]The figure showing MPEG image sequences (m= 3).
[Drawing 95]The figure showing what the storing output of the "I" picture is carried out.
[Drawing 96]The figure showing what the storing output of the "P" picture is carried out.
[Drawing 97]The figure showing what the storing output of the "B" picture is carried out.
[Drawing 98]The figure showing a picture ** buffer.
[Drawing 99]The figure showing H.261 prediction form.
[Drawing 100]The figure showing an H.261 series.
[Drawing 101]The figure showing the hierarchy of H.261 syntax.
[Drawing 102]The figure showing an H.261 image layer.
[Drawing 103]The figure showing H.261 arrangement of block groups.
[Drawing 104]The figure showing an H.261 "slice" layer.
[Drawing 105]The figure showing H.261 arrangement of a macro block.
[Drawing 106]The figure showing an H.261 block series.
[Drawing 107]The figure showing an H.261 macro block layer.
[Drawing 108]The figure showing H.261 arrangement of a pixel.
[Drawing 109]The figure showing the hierarchy of MPEG syntax.
[Drawing 110]The figure showing an MPEG series layer.
[Drawing 111]The figure showing the image layer of an MPEG group.
[Drawing 112]The figure showing an MPEG image layer.
[Drawing 113]The figure showing an MPEG "slice" layer.
[Drawing 114]The figure showing the block of an MPEG series.
[Drawing 115]The figure showing an MPEG macro block layer.
[Drawing 116]The figure showing the example of "open GOP."
[Drawing 117]The figure showing the example of access structure.
[Drawing 118]The figure showing access start timing.
[Drawing 119]The figure showing a high-speed page read-out cycle.
[Drawing 120]The figure showing a high-speed page write cycle.
[Drawing 121]The figure showing a RIFURESHU cycle.
[Drawing 122]The figure showing how a line and a column address are extracted from a chip address.
[Drawing 123]The figure showing the timing parameter for strobe signals.
[Drawing 124]The figure showing the timing parameter between which two strobe signals.
[Drawing 125]The figure showing the timing parameter between a bus and a strobe signal.
[Drawing 126]The figure showing the timing parameter between a bus and a strobe

signal.
[Drawing 127]The figure showing the Huffman decoder and a parser.
[Drawing 128]The figure showing H.261 and an MPEG AC coefficient decryption flow chart.
[Drawing 129]The block diagram of JPEG (AC and DC) coefficient decryption.
[Drawing 130]The figure showing the flow chart of JPEG (AC and DC) coefficient decryption.
[Drawing 131]The figure showing the interface of the Huffman token formatter.
[Drawing 132]The block diagram of a token formatter.
[Drawing 133]The figure showing H.261 and MPEG AC coefficient decryption.
[Drawing 134]The figure showing the interface of Huffman ALU.
[Drawing 135]The figure showing the essential structure of Huffman ALU.
[Drawing 136]The figure showing a buffer manager.
[Drawing 137]The block diagram of imodel and hsppk.
[Drawing 138]The figure showing an imex state.
[Drawing 139]The figure showing buffer start-up.
[Drawing 140]The figure showing a DRAM interface.
[Drawing 141]The figure showing a write-in swing buffer.
[Drawing 142]The figure showing an arithmetic block.
[Drawing 143]The block diagram of iq.
[Drawing 144]The figure showing an iqca state machine.
[Drawing 145]The figure showing IDCT 1-D conversion algorithms.
[Drawing 146]The figure showing IDCT 1-D conversion structure.
[Drawing 147]The block diagram of a token stream.
[Drawing 148]A standard block diagram.
[Drawing 149]The block diagram showing microprocessor test access.
[Drawing 150]The figure showing 1-D conversion micro structure.
[Drawing 151]The block diagram of a time decoder.
[Drawing 152]The figure showing the structure of 2 wiring interface stage.
[Drawing 153]The block diagram of an address generator.
[Drawing 154]The figure showing a block and offset of a pixel.
[Drawing 155]The figure showing two or more prediction filters.
[Drawing 156]The figure showing a prediction filter.
[Drawing 157]The figure showing a 1-D prediction filter.
[Drawing 158]The figure showing a 1-block pixel.
[Drawing 159]The figure showing the structure of a read-out rudder.
[Drawing 160]The figure showing a block and offset of a pixel.
[Drawing 161]The figure showing the example of prediction.
[Drawing 162]The figure showing a read-out cycle.
[Drawing 163]The figure in which writing and showing a cycle.

[Drawing 164]The block diagram of the top-level register group which can refer to timing.
[Drawing 165]The figure showing the control to which a presentation number is made to increase.
[Drawing 166]The figure showing a buffer manager State machine.
[Drawing 167]The figure showing the main loop of the State machine.
[Drawing 168]The figure showing the buffer 0 containing SIF (macro block of 22x18).
[Drawing 169]The figure showing the SIF ingredient 0 with a display window.
[Drawing 170]The figure showing the example of a graphics format which shows a storing block address.
[Drawing 171]The figure showing the buffer 0 containing SIF (macro block of 22x18).
[Drawing 172]The figure showing the example of address computation.
[Drawing 173]The figure showing a write-in address generation state.
[Drawing 174]The figure showing the slice of a data path.
[Drawing 175]The figure showing two-cycle operation of a data path.
[Drawing 176]The figure showing filtering in the mode 1.
[Drawing 177]The figure showing a level rise sampler data path.
[Drawing 178]The figure showing the structure of a color-space converter.
[Description of Notations]
33 [-- An action identification part, 41 / -- An output latch, 43, 44 / -- Register.] -- A token decoder, 34 -- An input latch, 36 -- A processing unit, 39